

トランジスタ技術

SPECIAL

No.45

特集 PC98シリーズのハードとソフト
386 & 486マシンを使いこなす



DESIGN WAVE

デザイン ウェーブ・シリーズは新世代エレクトロニクス・エンジニアのための設計ツール・ソフトウェアです。

大好評発売中!

パソコン用
電子回路シミュレータ

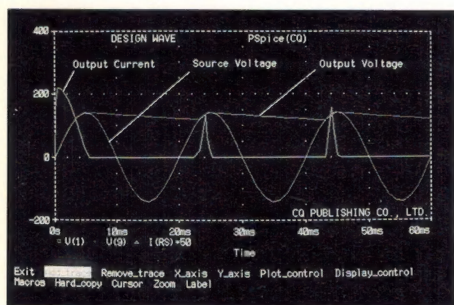
PSpice (CQ版) Ver.5

PC9801用 定価11,640円 送料 390円

PC/AT, J-3100用 定価14,640円 送料 390円

好評のPSpice (CQ版) が新しくなります。

従来からのPC9801シリーズ用に加えて、新たにPC/AT, J-3100用を用意しました。



- 回路の試作と実験を繰り返し行うのではなく、パソコンやワークステーション上で動作する電子回路シミュレータを駆使し、設計や解析を行う。これが新世代のエレクトロニクス・エンジニアのスタイルです。あなたも、パソコン用電子回路シミュレータPSpice (CQ版) を使って、新世代の設計技術をマスターしてみませんか。
- 可能な解析はトランジェント解析、AC解析、DC解析などのほか、フーリエ解析、モンテカルロ解析、感度解析、パフォーマンス解析などです。
- このPSpice (CQ版) には、この回路シミュレータを活用するためのバッチ・プログラムや、デバイス・ライブラリ、回路ファイル集、半導体デバイスをモデリングするためのツール、エディタ、各種ユーティリティを収めたオリジナル・ディスクが付属しています。

デザインウェーブ・ブックス①

パソコンCAD Future Net (CQ版) で実践する

電子回路図エディタ活用マスター

5インチ 2HD フロッピー×2枚付き 倉重 克己 著

B5変形判 271頁 定価3,000円(税込) 送料380円

重版出来! 好評発売中

●回路図エディタって何?

あなたは回路図をどうやって描いていますか? 手描きですか? それともお絵描きCADで? 回路図の清書が目的なら、お絵描きソフトと電子回路部品ライブラリがあれば十分でしょう。

でも、お絵描きソフトで描いた回路図と、回路図エディタで描いた回路図には決定的な違いがあるので。

回路図エディタで描いた回路図には、電気的な情報(ネット情報)が含まれているという点です。回路図ファイルから、ネット・リストを抽出し、回路シミュレータやプリント基板CADに渡したり、設計チェックを行ったり、部品表を作成したりすることができます。

本書にはFutureNet-62 (CQ版) が添付されており、回路図エディタの基本を、使いながら学ぶことができます。



《FutureNet (CQ版) の概要》

- ▶対応機種: PC9801, IBM-PC/ATおよび富士通FMRシリーズ
- ▶MS-DOS 3.0以降
- ▶RAM640Kバイト以上
- ▶プリントアウトOK, 最大50Kバイトのファイル・リード/ライトOK。
- ▶2FDシステムでも使えますが、ハード・ディスクの使用をおすすめします。

トランジスタ技術

CONTENTS

SPECIAL No.45

特集 PC98シリーズのハードとソフト

386&486マシンを使いこなす

吉田 功

第1章 PC98シリーズのシステム構成	2
98シリーズの違い	2
CPU	3
メモリ・マップ	5
I/Oポート	11
PC9801シリーズの割り込み	19
システム共通領域	22
コラム 機種別のI/Oポートのアクセス時間	10
第2章 ハードウェアの詳細	26
割り込みコントローラ	26
DMAコントローラ	34
タイマ	43
カレンダー時計	47
システム・ポート	50
キーボード・インターフェース	54
CRTディスプレイ	60
GDCと周辺LSI	65
GDC(μ PD7220)	70
フロッピー・ディスク・インターフェース	80
ハード・ディスク・インターフェース	83
マウス・インターフェース	91
プリンタ・インターフェース	94
RS-232-Cインターフェース	96
拡張RS-232-Cインターフェース	101
コラム システム共通領域	46
8255について	55
第3章 拡張スロットの信号と使い方	102
PC98シリーズの拡張スロットの詳細	102
拡張スロットの信号線	103
拡張スロットの電気的仕様	109
拡張基板の設計	114
第4章 PC98シリーズのBIOS	119
キーボードBIOS	119
CRT BIOS	121
グラフィックBIOS	123
RS-232-C BIOS	126
プリンタBIOS	128
DISK BIOS	129
ハード・ディスクBIOS	133
1MB/640KB両用フロッピー・ディスクBIOS	137
タイマBIOS	138
グラフィックLIO	140
C言語によるグラフィック操作	140
第5章 AD78090-4を使った拡張基板の製作	148
A-Dコンバータの製作	148
サンプル・プログラム・ディスク頒布のご案内	117

1 個別半導体素子 活用法のすべて 基礎からマスタするダイオード、トランジスタ、FET の実用回路技術	16 A-D/D-A の変換回路技術のすべて アナログとディジタルを結ぶ最新回路設計ノウハウ	31 基礎からのビデオ信号処理技術 複合映像信号の理解からハイビジョン信号の捉え方まで
2 作りながら学ぶ MC 68000 16ビットMPUとその周辺LSIを使いこなすためのハード&ソフト	17 OP アンプによる回路設計入門 アナログ回路の誤動作とトラブルの原因を解く	32 実用電子回路設計マニュアル アナログ回路の設計例を中心に実用回路を詳述
3 PC 9801 と拡張インターフェースのすべて 16ビット・パソコンを使いこなすためのハード&ソフト	18 ホビー・エレクトロニクス入門 売切	33 オプト・デバイス応用回路の設計・製作 光素子を使いこなすための製作ドキュメント
4 C-MOS 標準ロジック IC 活用マニュアル 実験で学ぶ 4000B/4500B/74HC ファミリー	19 PC 9801 計測インターフェースのすべて オリジナル拡張ボードでパソコンを実践活用しよう	34 つくる IC エレクトロニクス 機能 IC を使って実用機器を作ろう
5 画像処理回路技術のすべて カメラとビデオ回路、パソコンと融合させる	20 アナログ回路シミュレータ活用術 ゲーム感覚の回路設計を体験しよう	35 C 言語による回路シミュレータの製作 Quick C でのプログラミングとフィルタ回路の解析
6 Z80ソフト&ハードのすべて 基礎からマクロ命令を使いこなすまでのノウハウを集大成	21 デジタル・オーディオ技術の基礎と応用 最もポピュラーな最新技術を理解しよう	36 基礎からの電子回路設計ノート トランジスタ回路の設計からビデオ画像の編集まで
7 HD64180徹底活用マニュアル Z80を超えた高性能8ビットCPUのすべて	22 デジタル回路ノイズ対策技術のすべて TTL/CMOS/ECL の活用法と誤動作/トラブルへの処方	37 実用電子回路設計マニュアルⅡ 豊富な回路設計例から最適設計を学ぼう
8 データ通信技術のすべて シリアル・インターフェースの基礎からモデムの設計法まで	23 回路デザイナーのための PLD 最新活用法 PLD のプログラミング法から PAL ライタの製作まで	38 Z 80 システム設計完全マニュアル 周辺 I/O ボードの設計とマイコン・システムの開発
9 パソコン周辺機器インターフェース詳解 セントロニクス/RS-232C/GPIB/SCSI を理解するために	24 C による組み込み機器用プログラミング 16ビットCPUによるメカトロニクス入門	39 A-D コンバータの選び方・使い方のすべて アナログ信号をディジタル処理するための基礎技術
10 IBM PC&80286 のすべて 世界の標準パソコンとマルチタスクの基礎を理解する	25 最新マイコン・メモリ・システム設計法 DRAM, SRAM の動作からデュアル・ポート RAM, FIFO の活用まで	40 電子回路部品の活用ノウハウ 機器の性能と信頼性を支える受動部品の使い方
11 フロッピー・ディスク・インターフェースのすべて 需要の急増するFDDシステムの基礎から応用まで	26 68000 ソフト&ハードのすべて 実用ライブラリの作成と便利チップ 68301/68303 の活用技術	41 実験で学ぶ OP アンプのすべて 汎用 OP アンプから高性能 OP アンプまで
12 入門ハードウェア 手作り測定器のすすめ 電子回路設計の基礎と実践へのアプローチ	27 ハードディスクと SCSI 活用技術のすべて 本格活用のためのハード&ソフトのすべてを詳解	42 高速ディジタル回路の測定とトラブル解析 ハイスピード・ディジタル信号を高周波と捉ええる
13 シミュレータによる電子回路理論入門 コンピュータを使ったアナログ回路設計の手法を理解するために	28 最新・電源回路設計技術のすべて 3 端子レギュレータから共振型スイッチング電源まで	43 C によるマイコン制御プログラミング 86 系ペリフェラルを中心とした
14 技術者のための C プログラミング入門 MS-C, Quick C, Turbo C によるソフトウェア設計のすべて	29 マイコン独習 Z 80 完全マニュアル 手作りの原点から実用ソフトの作成まで	44 フィルタの設計と使い方 アナログ回路のキーポイントを探る
15 アナログ回路技術の基礎と応用 計測回路技術のグレードアップをめざして	30 ニュー・メディア時代のデータ通信技術 赤外線、無線通信技術から LAN, 光ファイバを用いた高速通信技術まで	年間購読のご案内 ●年間購読のお申し込みは購読料 10,860 円(税、送料込み)を添えて現金書留または郵便振替で、下記営業部へお申し込みください。

特集 PC98 シリーズのハードとソフト

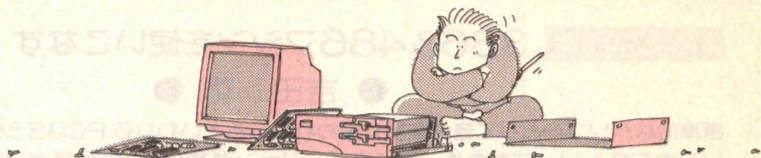
386&486マシンを使いこなす

● 吉田 功 ●

標準的なパソコンとして最も数多く使われているといわれる PC98 シリーズには、エプソンの PC シリーズを含め、多くのバージョンが存在し、新機種が発売され続けています。基本的なシステム構成は変わらないものの、使う側にとってバージョンの違いを知ることが重要です。ここでは最新の 98 シリーズのハードとソフトを詳解しています。



1 PC98シリーズのシステム構成



98 シリーズの違い

1982年の秋に、NECから初めてのPC98シリーズであるPC9801が発表されて以来、現在までに非常に多くの機種が発売されてきました。1987年の春には、EPSONからも98互換機としてPC98シリーズと、ほぼ同機能なコンピュータEPSON-PCシリーズが発表され、NEC同様に多くの機種が発売されています。

現在、PC98シリーズは目的別に数種類に分かれます。まず、シリーズの核になっているのはPC9801系列です。ノーマル・モードと呼ばれます。普通、98といえば、この系列のことをさしますし、他の系列の機種でもノーマル・モードを持っています。

次に、そのPC9801シリーズを携帯用に小型化したPC9801ノート・ラップトップ系列で、PC9801のソフトウェアをそのまま動作できるように作られていますので、かなりの部分で互換性があります。例外はPC98LT/HAで、PC9801系列とは画面表示関係が大きく異なり、互換性が少ないものになっています。

また、狭い画面表示を補うために1120×750ドットの表示が可能になったハイレゾ対応機種が1985年に生まれました。名前がPC9801ではなくPC98と呼ばれて差別化されています。初めてのハイレゾ対応機であるPC98XA以外の機種では、ハイレゾとノーマルの二つのモードを持ちますので、ノーマル・モードの多くのソフトウェアを、そのまま使用することができます。また、1990年からはPC-H98シリーズと独立したシリーズになりました。

1992年からは、PC9821と呼ばれる新しいシリーズができました。640×480ドット256色の画面を持つシリーズで、マルチメディアや、MS-WINDOWSの使用に有利なように作られています。基本的にはPC9801シリーズと同等で、新しい機能が追加される形になっています。

これらの新しい機能は具体的にハードウェアを直接操作せずに、BIOS等を通して操作するのを前提としているようで、同じ9821の名前が付いていても、ハ

ード的には異なる場合があるようです。

◆ ノーマル・モードとハイレゾ・モード

ノーマル・モードとハイレゾ・モードの違いは、表示画面サイズが変わっただけではなく、メモリ・マップも大きく変更されています。

まず、メイン・メモリ空間(コンベンショナル・メモリ)が128Kバイト増えて768Kバイトになりました。VRAM空間もC0000H~EFFFFH(テキストはE0000H~E3FFFFH)になり、一箇所にまとまっています。

I/O関係も若干異なり、マウス・インターフェース等ではアドレスも異なります。プリンタ・インターフェースでは、フル・セントロニクス仕様に拡張されているためにI/Oアドレスは同じものの、内容は大きく違います。他にも細かい部分で、変更されています。

ハイレゾ・モードは、ノーマル・モードより、いろいろなところで優れていますが、NECの販売戦略のためか、価格が全般的に高く設定され、一般的ではありません。筆者もまた、ハイレゾ・マシンには縁遠いためもあって、本誌では、ハイレゾ・モードについては詳しく述べることはできませんでした。

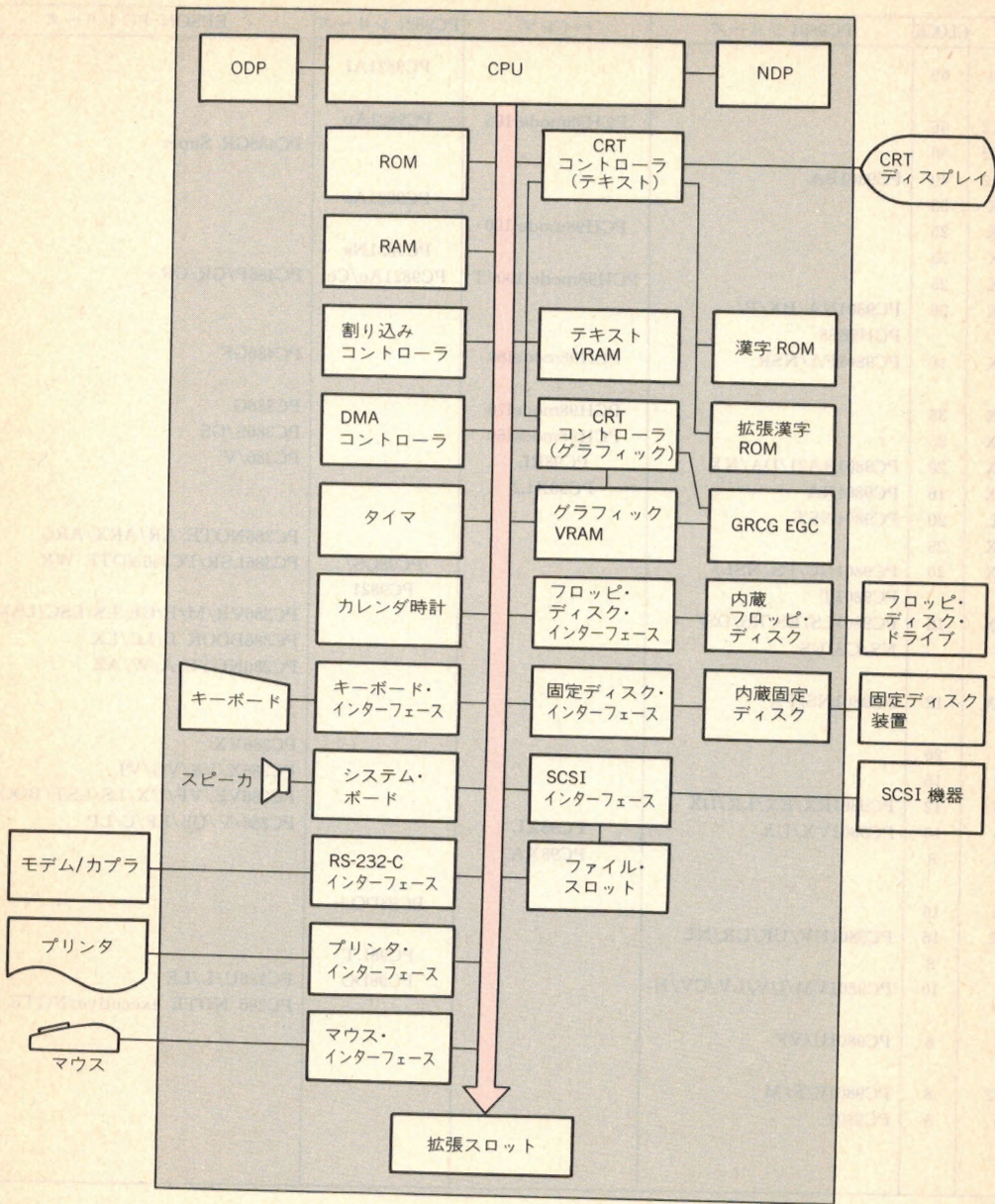
◆ 周辺LSIの違い

PC98シリーズのCPUは、年々新しく違ったものが搭載されてきましたが、CPUの高速化に合わせて、周辺LSIも、図1-1に示すようにグレードアップされてきました。搭載されているLSIは、細かい追加

〈図1-1〉周辺LSIの移り変わり

LSI	主に 386 までの機種	主に 486 以降の機種
	NMOS	CMOS
DMAC	μPD8237AC	→ μPD71037
SIO	μPD8251AC	→ μPD71051
PIT	μPD8253C	→ μPD71054
PPI	μPD8255AC	→ μPD71055
PIC	μPD8259AC	→ μPD71059
FDC	μPD765A	→ μPD72065
GDC	μPD7220	→ μPD72020

〈図 1-2〉 PC98 シリーズのハードウェアのブロック・ダイアグラム



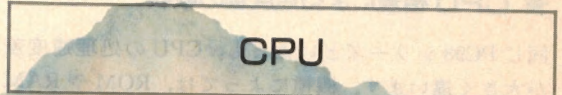
機能があつたり、処理速度の向上がなされていますが、完全上位互換になっています。

LSI の処理速度の向上は、I/O のリカバリ・タイムの短縮に係りてきます。μPD710××を載せている機種では、CPU 速度が上がったにも関わらずリカバリ・タイムが減っています。

また、最近の機種では、周辺 LSI のほとんどが集合化されカスタム・チップになってきました。μPD71055 等の汎用 LSI は、メイン基板からなくなり、CPU とメモリ以外はいくつかのカスタム LSI が載っているだけになっています。

◆ ブロック・ダイアグラム

PC98 シリーズのハードウェアをブロック図で表してみました。機種によって搭載されている機能が、かなり変わりますが、基本的には図 1-2 のようになります。



PC98 リーズには、非常に多くの機種があり、それに搭載されている CPU も数多くあります。最初に

〈図 1-3〉 PC98 シリーズに搭載されている CPU

CPU	CLOCK	PC9801 シリーズ	ハイレゾ	PC9821 シリーズ	EPSON-PC シリーズ
Pentium	60			PC9821Af	
80486DX2	66		PCH98model105	PC9821Ap	PC486GR Super
80486DX2	50				
80486DX2	40	PC9801BA			
80486DX	33			PC9821As	
80486DX	25		PCH98model100		
80486SX	33			PC9821Ne	
80486SX	25		PCH98model190/T	PC9821Ae/Ce	PC486P/GR/GR+
80486SX	20	PC9801NA/BX/P/ PCH98S8			
80486SX	16	PC9801FA/NSR	PCH98model80		PC486GF
80386DX	33		PCH98model70		PC386G
80386DX	25		PCH98model60		PC386S/GS
80386DX	20	PC9801RA21/DA/NX	PC98RL		PC386/V
80386DX	16	PC9801RA	PC98XL2		
80386SL	20	PC9801NST			
80386SX	25				PC386NOTE AR/ARX/ARC
80386SX	20	PC9801NC/FS/NSL/ PC9801T		PC98GS/ PC9821	PC386LSR/PC386NOTE WR
80386SX	16	PC9801LS/ES/RS/DS/ NS/CS/US			PC386VR/M/P/GE/LS/LSC/LSX PC386BOOK L/LC/LX PC386NOTE A/W/AE
80386SX	12	PC9801NS/FX			
80286	20				PC286VX
80286	16				PC286X/VS/VG/VJ
80286	12	PC9801RX/EX/LX/DX			PC286VE/VF/UX/LS/LST/BOOK
80286	10	PC9801VX/UX	PC98XL		PC286/V/US/LF/C/LP
80286	8		PC98XA		
V33A	16			PC98DO+	
V30HL	16	PC9801NV/UF/UR/NL			
V50	8			PC98LT	
V30	10	PC9801VM/UV/LV/CV/N		PC98DO	PC286U/L/LE PC286 NOTE executive/NOTE F
V30	8	PC9801U/VF			
8086-2	8	PC9801E/F/M			
8086	5	PC9801			

登場した PC9801 に採用された 8086 から、最新の Pentium まで、その実行速度の差は数十倍にも達します。搭載 CPU 別に機種名をまとめてみました(図 1-3)。

◆ CPU 格差による速度差の吸収

同じ PC98 シリーズといえども、CPU の処理速度差が大きく違います。機種によっては、ROM や RAM の速度が CPU に付いてきませんし、拡張スロットの規格は 80286 の 10 MHz の頃から変わっていません

から、CPU がメモリや周辺に合わせてウェイトをかけます。

ウェイトの数は、大まかに分けると図 1-4 に示すように、CPU の種類とクロックの周波数で分けられます。機種によって若干違いがあるものがあります。

◆ 内蔵 RAM と増設 RAM

本体のマザー・ボード自体に初めから付いている内蔵 RAM と、後からメモリ専用スロットに増設する専用拡張 RAM、本体後部の拡張スロットに増設する汎

〈図 1-4〉 CPU の種類とクロック周波数で分けるウェイト

8086	5MHz	8MHz
RAM/ROM	0	1
I/O	1	2

9801/E/F/M

V30	8	10	10	16
内蔵 RAM	1	1	0	0
拡張スロット ROM	2	3(2) UV21	2	6(1) NV, NL
I/O	2	3	3	8(2)

9801U/VF/VM/UV LV/CV/UV/N UF/UR/NV/NL

0.24 2 21 21 11

21 21 22

286	8	10	12
内蔵 RAM	0	0	0
拡張 RAM	1	1	0(専用)
拡張スロット ROM	4	5	6
I/O	3	4	6

386	12	16	20
内蔵 RAM	0	0	0
専用拡張 RAM		0	0
拡張スロット ROM	5	8-10	10-12
I/O	5-6	8	10

486	16MHz 9801FA	20	25	33
内蔵 RAM	0	2/1	2/1	2/1
専用拡張 RAM	1	2	2	2
拡張スロット ROM	9	11	14?	19?
I/O	9	11	14?	19?

用拡張 RAM とがあります。これらは自動的に挿入されるウェイトが変わります。特に CPU に 80386 を搭載している場合は、プロテクト・メモリを有効利用できますので、ウェイト数が少ない専用拡張 RAM が有利です。

◆ CPU に対する命令

80286 以上の CPU を搭載している機種には、CPU に対してのみ行う CPU リセットと、プロテクト・モード ON の操作を行うポートがあります。

CPU リセットは、CPU のみにリセットをかけて、周辺の LSI などへはそのままです。80286 には、8086 互換モードのリアル・モードと、1M バイト以上のメモリ空間を使用することができるプロテクト・モードがありますが、プロテクト・モードからリアル・モード

への移行はソフトウェアではできません。そのため CPU にリセットをかけることで、これを実現しています。80386 以降の CPU では、ソフトウェアでこれらの切り替えができます。

CPU にリセットをかける場合、電源投入時なのか、リセット・スイッチが押されたのか、モード切り替えのために CPU のみにリセットをかけたのかは、システム・ポートのポート C・D₅/D₇を参照すれば知ることができます。

プロテクト・モード ON は、8086 や V30 等のメモリ空間が 1M バイトしかない機種と、80286 以上のメモリ空間が 1M バイト以上ある機種との違いを吸収するもので、プロテクト・モード ON にすることで、1M バイト以上のメモリをアクセスすることができるようになります。

80286 等の 1M バイトを超える CPU では、8086 互換モードでもアドレス・ラインは 1M バイト以上のメモリをアクセスできます。例えば、セグメント = FFFFH/オフセット = FFFFH とすれば、ソフトウェア的には、FFFFFH + OFFFHH = 10FFFEFH と、1M バイト + 64K バイトのアクセスが可能となります。

しかし、8086/V30 では、アドレス・ラインが 20 本しかないため、10FFFEFH とはならず、OFFFEFH となってしまいます。

プロテクト・モード ON は、これらの違いを吸収するために付けられています(図 1-5)。

PC9801RA21/DA/FA 等の機種では、プロテクト・モード ON に加えてプロテクト・モード OFF(図 1-6)も可能です。

◆ プロセッサ ID

80386 以降の CPU では、CPU がリセットされた直後に DX レジスタにプロセッサ ID が入っています。機種によっては、この値をシステム共通領域の 0000:0486H-0487H に書き込まれます。80286 等の CPU では違うデータが入っているようです。

CPU リセットとシステム・ポートのポート C・D₅/D₇を操作することで、プログラム中からも、プロセッサ ID を読み出すことができます(図 1-7)。

メモリ・マップ

PC98 シリーズは、8086 をベースにして拡張されてきました。CPU 自身は、8086 では 1M バイトを、80286 では 16M バイト(仮想アドレスは 1G バイト)、80386/80486 では 4G バイト(仮想アドレスは 64T バイト)のメモリを管理できるのですが、PC98 シリーズでは大まかに分けると 3 種類のメモリ・マップに分けられます。

〈図 1-5〉 CPU に対する命令

命 令	I/O アドレス	R/W	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	機能
CPU Reset	00F0H	W	0	0	0	0	0	0	0	0	CPU, NDP の初期化。 復旧できないエラーの対応, プロ テクト・モードからリアル・モー ドへの移行に使用。
プロテクト・モード ON	00F2H	W	0	0	0	0	0	0	0	0	アドレス・バス上位 4 ビットのマ スクを解除し 100000H 以上のメ モリをアクセス可能にする。

〈図 1-6〉 プロテクト・モード OFF

命 令	I/O アドレス	R/W	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	機能
プロテクト・モード ・コントロール	00F6H	W	0	0	0	0	0	0	0	WAEN	WAEN を 1 にすることにより, 100000H 以上のアドレスがアク セス不可になる。

◆ 基本のメモリ・マップ

基本のメモリ・マップは 8086 を搭載した初代の PC9801 と同様の 1M バイトです。この 1M バイトのメモリ・マップも大きく分けると次の 3 種類の領域に分けられます。

- (1) 00000H～9FFFFH までは RAM の空間
- (2) A0000H～BFFFFH および
E0000H～E7FFFFH までは VRAM 空間
- (3) C0000H～DFFFFH までは拡張用の ROM 空間
E8000H～FFFFFH まではシステム用の ROM 空間

メモリ・マップを図 1-8 に示します。

システム ROM と RAM の空間は各機種とも共通で、古い機種では RAM が最初から 640K バイト分 (00000H～9FFFFH) 載っていない機種もありますが、拡張スロットに RAM を増設することで 640K バイトまで増やせます。また、RAM が最初から 640K バイト以上搭載されている機種では、ディップ・スイッチでバンク 8～9 (80000H～9FFFFH) の内蔵 RAM を殺すことができます。これは、バンク切り替え式の RAM ボードを拡張スロットに使用する場合に使います。

◆ 拡張 ROM 空間

主にインターフェース等の拡張ボード用の ROM 空間で使われています。

拡張 ROM 空間は、ユーザ用拡張 ROM 空間 (C0000H～C7FFFFH) とシステム用拡張 ROM 空間 (C8000H～DFFFFH) に分かれます。システム用 ROM 空間は一般的にはメーカーから供給されるインターフェース・ボード (FDD や HDD 等) の基本的なボードの ROM 空間です。

拡張ボード用の ROM 空間は、図 1-9 に示すよう

〈図 1-7〉 CPU の識別

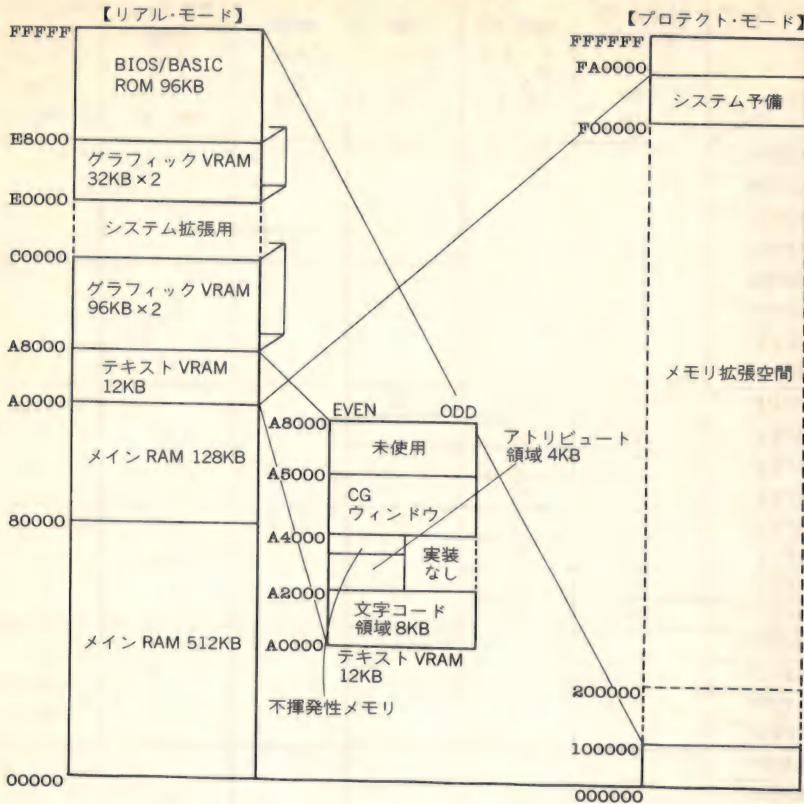
プロセッサ名	コンポーネント ID 〈DH レジスタ〉	ステッピング/ リビジョン ID 〈DL レジスタ〉
80386DX/SX	×3H	××H
80486DX	04H	0×H
80486DX50	04H	1×H
80486SX	04H	2×H
80487SX・MCP	04H	2×H
80486DX2	04H	3×H
80486DX・ODP	04H	3×H
80486SX・ODP	04H	3×H

にそのボードによって使用されるアドレスの標準的な位置が決まっています。中には ROM のアドレスを可変できるものもありますが、アドレスが固定されているものもあります。

また、**拡張ボードではなく、本体自身でこの空間に ROM を持ってる機種があります。** 代表的なものには、FM 音源ボード用の ROM や、HDD インターフェース用の ROM があります。さらに、ノート等には独自の仕様に基づいてシステム用の RAM 空間や、レジューム用に使われている機種もあります。

拡張ボード用の ROM 空間に、RAM をマッピングして EMS メモリとして使用されることもあります。CPU が 80286 以下の機種では、拡張スロット上の EMS 対応 RAM ボードで、この領域に RAM をハードウェア的にマッピングして使用します。80386 以上の CPU の機種では、CPU 自身が持つメモリ・マネージメントの機能を使用して、プロテクト・メモリをこの領域にマッピングして使用します。

〈図 1-8〉 基本メモリ・マップ(ノーマル・モード)



● 共通事項

- ・ 80000H-9FFFFH は DIP-SW₃₋₆ で RAM から切り離せる。
- ・ A3FE2H, A3FE6H, A3FEAH, A3FEEH, A3FF2H, A3FF6H, A3FFAH, A3FFE2H, A3FE6H, A3FEAH, A3FEEH, A3FF2H, A3FF6H がメモリ・スイッチとして使用される。

● EPSON 共通仕様

- ・ DIP-SW₃₋₆ が OFF (ノーマル・モード) のときは、0 除算エラー例外 (割り込みベクタ 00H)、セグメント・オーバ・ラン (割り込みベクタ 0DH) に対応するアドレス (0: 0000H-0003H, 0: 0034H-0037H) はシステムで使用しており、ソフトウェアでライトしたデータと、リードしたデータが一致しない。このためワーク・エリアとして使用できない。

● 機種別仕様

- ・ プロテクト・メモリは 286 以降の CPU でプロテクト・モード時のみアクセス可能である。
- ・ 0000: 0400H-0000: 05FFFH はシステム共通域。システム共通域はソフトウェアからの参照のみを行い、情報の更新は行わないこと。
- ・ EMS 対応の機種では、B0000H-BFFFFH の 64KB の空間に EMS ページ・フレームの割り当てが可能。
NEC: PC9801RA 以降で 80286CPU 以上の搭載機
EPSON: 286VX, 286VG, 386V, 386VR, 386S, 386G, 386GS, 386GE, 486GR, 486GF
- ・ CG ウィンドウ (A400H-A4FFFH) が搭載されていない機種。
NEC: PC9801/E/F/M/U/VF/VM/UV/LV/CV/NV, PC98LT
EPSON: 286, 286V, 286VE, 286U, 286L

● NEC 機種別仕様

- ・ 1000000H 以降のメモリは PC9801Af でアクセス可能である。
- ・ PC9801VX01, 21, 41 では高速グラフィック処理用に ROM を 2 組実装。80286CPU/10MHz モード時に裏 ROM を使用する。
- ・ GVRAM1~3, 同一アドレスに VRAM96KB を 2 組実装。PC9801U2 は一組。
- ・ GVRAM4 は 16 色グラフィック対応 VRAM であり、16 色未対応機では実装されていない。PC9801U2 では 32KB 一組。

〈図 1-9〉 ROM 搭載オプション・ボードのアドレス空間

オプション・ボード アドレス空間		640KB FD インター フェース -08 -09	1MB FD インター フェース -15	HD インター フェース -27	SCSI インター フェース -55	RS-232-C 拡張 インター フェース 9861/K	サウンド・ インター フェース U-03 -26/K	専用 HDD (IDE) および RAM ドライブ
シ ス テ ム 予 約	DF000~DFFFF							
	DE000~DEFFF							
	DD000~DDFFF					○		
	DC000~DCFFF				◎			
	DB000~DBFFF							
	DA000~DAFFF					○		◎
	D9000~D9FFF							
	D8000~D8FFF							
	D7000~D7FFF	○	◎	◎				
	D6000~D6FFF	◎	○			○		
	D5000~D5FFF	○	○					
	D4000~D4FFF	○	○					
	D3000~D3FFF	○	○					
	D2000~D2FFF	○	○			◎		
	D1000~D1FFF	○	○					
	D0000~D0FFF	○	○					
	CF000~CFFFF							
	CE000~CEFFF					○		
	CD000~CDFFF						◎	
	CC000~CCFFF							
ユ ー ザ 解 放 R O M 空 間	CB000~CBFFF							
	CA000~CAFFF					○		
	C9000~C9FFF							
	C8000~C8FFF							
	C7000~C7FFF							
	C6000~C6FFF					○		
	C5000~C5FFF							
	C4000~C4FFF							
	C3000~C3FFF							
	C2000~C2FFF							
	C1000~C1FFF					○		
	C0000~C0FFF							

◎：工場出荷時設定 ○：変更可能アドレス

◆ VRAM 空間

VRAM 空間は、テキスト VRAM とグラフィック VRAM の 2 種類があります。グラフィック VRAM は、古いデジタル RGB・8 色専用の機種では、A8000H~BFFFFFFH の 96K バイトですが、アナログ RGB 対応・16 色の機種では、それに加えて、E0000H~E8000H の 32K バイトが拡張されて、合計 128K バイトあります。

テキスト VRAM 空間は A0000H~A3FFFFH までですが、細かく分けると、文字データ用、アトリビュート用、不発揮メモリ、CG ウィンドウ(A4000H

~A4FFFFH)等に割り当てられています。

◆ 80286 以上のメモリ・マップ

CPU に 80286 以上を持つ機種の場合は、1M バイト以上のメモリを使用できます。ただし、MS-DOS を使用する場合は、80386 以上の CPU がないとプロテクト・メモリを有効に使用することができません。

メモリの上限は、DMA のアクセス範囲が 16M バイトのために、これに制限されています。また、メモリ空間上端の F00000H~FFFFFFFH(1M バイト)は、ROM 空間のイメージ等があり使用されないで、実際に使用できる RAM 空間は、640K バイト(コン

〈図 1-10〉 I/O ポート

0 1 2 3 4 5 6 7								8 9 A B C D E F								0 1 2 3 4 5 6 7								8 9 A B C D E F																																	
00	8259 マスタ							01	8237 DMAC							80	SASI HDD							81	予約																																
02	イメージ							03								82	予約							83								BRANCH 4670							ネット ワーク 1F																		
04								8259 スレープ								84								765 IMB・FDD																																	
06								イメージ								86																														IMB・FDD											
08	8259 スレープ															88															イメージ																										
0A	予約							8C								7220 スレープ																																									
0C								イメージ															8E	グラフィック表示																																	
0E	予約							90															通信制御アダプタ or 拡張 RS-232-C																																		
10								4990 カレンダー																							92	IBM/640KB・FDD 切り替え																									
12	イメージ							94								765 640KB・FDD																																									
14								8251 RS-232-C							96															640KB・FDD																											
16								イメージ							98								ユーザ開放																																		
18	8251 RS-232-C														9A																						ユーザ開放																				
1A	イメージ							9C	ユーザ開放																																																
1C								8251 RS-232-C								9E	ユーザ開放																																								
1E	予約							AO								ユーザ開放																																									
20								4990 カレンダー																8A	ユーザ開放																																
22	イメージ							8C	ユーザ開放																																																
24								DMA バンク・レジスタ															8E	ユーザ開放																																	
26								イメージ								90	ユーザ開放																																								
28	イメージ															92															ユーザ開放																										
2A	イメージ							94	ユーザ開放																																																
2C								DMA バンク・レジスタ								96								ユーザ開放																																	
2E	イメージ							98								ユーザ開放																																									
30								8251 RS-232-C															9A								ユーザ開放																										
32	イメージ							9C	ユーザ開放																																																
34								システム・ポート															9E	ユーザ開放																																	
36								イメージ								80	ユーザ開放																																								
38	439 DMA・MASK															82															ユーザ開放																										
3A	イメージ							84	ユーザ開放																																																
3C								キャッシュ・フラッシュ								86								ユーザ開放																																	
3E	イメージ							88								ユーザ開放																																									
40								43F															8A								ユーザ開放																										
42	8251 キーボード							8C	ユーザ開放																																																
44								イメージ															8E	ユーザ開放																																	
46																8251 キーボード							90															ユーザ開放																			
48	イメージ							92								ユーザ開放																																									
4A								8251 キーボード							94								ユーザ開放																																		
4C	イメージ							96	ユーザ開放																																																
4E								8251 キーボード																						98	ユーザ開放																										
50	NMI コントロール							9A								ユーザ開放																																									
52	イメージ							9C															ユーザ開放																																		
54								イメージ							9E															ユーザ開放																											
56															8255 320KB・FDD																						80	ユーザ開放																			
58	イメージ							82	ユーザ開放																																																
5A								8255 320KB・FDD								84	ユーザ開放																																								
5C	タイム・スタンパ							86								ユーザ開放																																									
5E								8255 320KB・FDD																88	ユーザ開放																																
60	7220 マスタ							8A	ユーザ開放																																																
62	テキスト表示							8C															ユーザ開放																																		
64								テキスト表示								8E	ユーザ開放																																								
66																7220 マスタ														90	ユーザ開放																										
68	テキスト表示							92	ユーザ開放																																																
6A								7220 マスタ								94								ユーザ開放																																	
6C	テキスト表示							96								ユーザ開放																																									
6E								7220 マスタ															98								ユーザ開放																										
70	52611 CRT コントローラ							9A	ユーザ開放																																																
72								52611 CRT コントローラ															9C	ユーザ開放																																	
74																52611 CRT コントローラ							9E															ユーザ開放																			
76	52611 CRT コントローラ							80								ユーザ開放																																									
78								52611 CRT コントローラ							82								ユーザ開放																																		
7A	GRCG・EGC							84	ユーザ開放																																																
7C								GRCG・EGC																						86	ユーザ開放																										
7E	GRCG・EGC							88								ユーザ開放																																									
	偶数バイト							8A															ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト														9E																						ユーザ開放																				
								偶数バイト							80	ユーザ開放																																									
	偶数バイト														82								ユーザ開放																																		
								偶数バイト							84															ユーザ開放																											
	偶数バイト														86																						ユーザ開放																				
								偶数バイト							88	ユーザ開放																																									
	偶数バイト														8A								ユーザ開放																																		
								偶数バイト							8C															ユーザ開放																											
	偶数バイト														8E																						ユーザ開放																				
								偶数バイト							90	ユーザ開放																																									
	偶数バイト														92								ユーザ開放																																		
								偶数バイト							94															ユーザ開放																											
	偶数バイト														96																						ユーザ開放																				
								偶数バイト							98	ユーザ開放																																									
	偶数バイト														9A								ユーザ開放																																		
								偶数バイト							9C															ユーザ開放																											
	偶数バイト																																																								

ベンショナル・メモリ)+14M バイト(プロテクト・メモリ)の 14.6M バイトになります。ただし、機種によっては、メモリ増設ボードの限界で、14.6M バイトまで増設できない場合があります。

◆ PC9821Afのメモリ・マップ

PC9821Af 以降の機種では、32 ビット(4G バイト)まで DMA でアクセスが可能なものがあります。これらの機種では、16M バイトを超える RAM の増設が

可能なものもあります。

アドレス空間はたくさんあっても、そのすべてに RAM を載せるのは不可能なようで、最近の機種ではメモリ増設ボードの増設限界でメモリの増設上限が決まってしまうようです。PC9821Af では 79.6M バイトが上限ですが、PC9821Ap2 等では 73.6M バイト、PC9821Bp 等では 35.6M バイトです。

機種別の I/Oポートのアクセス時間

I/O のアクセス時間は、本来、CPU の命令実行時間だけのはずですが、実際には、周辺 LSI が拡張スロットと同じバスに接続されているために、拡張スロットのクロック(システム・クロック)と同じになるまで、CPU にウェイトを入れています。

しかし、実際に I/O アクセス時間を計測してみると、システム・クロックのウェイト時間以上に速度が遅くなります。つまり、必用以上のウェイトをハードウェアが故意に挿入していることになります。

● I/O リカバリ・タイム

これは遅い周辺 LSI(RS-232-C 用の 8251 等)にアクセスする場合に、周辺 LSI の処理速度より、CPU のアクセス時間のほうが早くなってしまうため、ハード的にウェイトを入れてソフトウェアでタイミング(リカバリ・タイム)を取らなくて良いようにという配慮らしく、EPSON の PC286/V/U/L/LE 以外の機種では、自動的にリカバリ・タイムが挿入されます。

実際にどのようにリカバリ・タイムが挿入されるか、測定用プログラムを作成して、測定してみた結果です。

〈図 1-A〉 デスクトップ(リアル・モード)

機種名 使用 CPU	実行速度 IN 上段 OUT 下段	8253・クロック NOP 実行速度	I/O アクセス時間(上段=IN, 下段=OUT)				
			30H	5CH	5FH	DOH	AOH
			RS-232-C	T.S	WAIT	FREE	GDC
PC-H98 改 m105 486DX2-80		1.9968MHz	877ns	879ns	878ns	1375ns	881ns
		15ns NOP	4513ns	884ns	858ns	1372ns	885ns
PC9801Ap 486DX2-66		2.4576MHz	863ns	864ns	864ns	1897ns	866ns
		19ns NOP	7397ns	894ns	1709ns	2540ns	894ns
PC9801As 486DX-33		2.4576MHz	1041ns	1042ns	1042ns	1898ns	1039ns
		30ns NOP	7396ns	1126ns	1710ns	2534ns	1125ns
PC9801BX 486SX-20		2.4576MHz	1346ns	1346ns	1347ns	2445ns	1348ns
		51ns NOP	7414ns	1439ns	1728ns	2440ns	1434ns
PC9801FA21 486SX-16		1.9968MHz	1528ns	1526ns	1530ns	2505ns	1529ns
		65ns NOP	4504ns	1647ns	1647ns	2506ns	1648ns
PC9801DA2 386-20	659ns	2.4576MHz	1253ns	1250ns	1252ns	2441ns	1249ns
	557ns	152ns NOP	4470ns	665ns	666ns	2439ns	665ns
PC9801RA21 386-20	667ns	2.4576MHz	1253ns	1251ns	1252ns	2441ns	1251ns
	565ns	154ns NOP	4469ns	666ns	667ns	2438ns	666ns
PC9801ES 386SX-16	810ns	1.9968MHz	1404ns	1404ns	1405ns	2503ns	1405ns
	686ns	187ns NOP	4504ns	763ns	763ns	2500ns	763ns
PC286VF 286-12	417ns	2.4576MHz	1178ns	1181ns	1181ns	4177ns	1180ns
	250ns	250ns NOP	2751ns	1224ns	1224ns	3005ns	1224ns
PC286VE 286-12	417ns	2.4576MHz	1187ns	1191ns	1188ns	3004ns	1186ns
	250ns	250ns NOP	2755ns	1231ns	1234ns	3004ns	1232ns
PC9801RX2 286-12	407ns	2.4576MHz	947ns	945ns	945ns	2519ns	945ns
	244ns	244ns NOP	4551ns	784ns	785ns	2523ns	782ns
PC9801VX21 286-8	507ns	2.4576MHz	960ns	958ns	959ns	2442ns	957ns
	304ns	304ns NOP	4474ns	753ns	753ns	2443ns	757ns
PC9801VM2 V30-10	867ns	2.4576MHz	1389ns	1392ns	1391ns	1389ns	1393ns
	867ns	325ns NOP	1387ns	1388ns	1388ns	1387ns	1391ns
H98 model 60/70					600ns		
H98 model 80/90/100					1000ns		

I/Oポート

PC98 シリーズで使用されている、周辺インターフェース LSI のデータ・バスの幅は 8 ビットのもので、拡張スロットのバス幅は 16 ビットですから、データ・バスを上位 8 ビットと下位 8 ビットの二つに分けて使用します。このため、上位につながれた LSI は奇数アドレスになり、下位につながれた LSI は偶

数アドレスに割り当てられます。したがって、LSI 自身が複数のアドレスを使用する場合は、CPU からは奇数・偶数の一つ飛びのアドレスに見えます(図 1-10)。

80386DX や 80486 等の CPU は 32 ビットのバスを持っています。しかし、拡張スロットのバス幅は 16 ビットですから、32 ビットのバスを上位 16 ビットと下位 16 ビットに分けて、16 ビット・バスとして使用する必要があります。これらの CPU ではバス・サイ

プログラムは、NOP, IN, OUT の命令を実行する時間を計測するものです。

この結果、RS-232-C やキーボード・インターフェース用の 8251 には、どの機種にもかなりのウェイトが挿入されているようです。D0H からのユーザに解放されている I/O アドレスも、他のアドレスと比べて余計にウェイトがかかる傾向があります。また、PC9801FA 以降の機種は、全体的にウェイトが多くなっています。

H98 シリーズや 80486 以上の CPU 以上では標準と

なった OUT 5FH, AL 命令も、機種によって結構時間が変わることもわかりました。

● 参考

対象となる命令の実行時間(クロック数)。

命令	CPU	クロック数		
		8086	80286	80386
IN AL, DX		8	5	13(27)
OUT DX, AL		8	3	11(25)
NOP		3	3	3

カッコ内は仮想 86 モード

〈図 1-B〉 ノート関連

機種名 使用 CPU	8253・クロック NOP 実行速度	I/O アクセス時間(上段=IN, 下段=OUT)				
		30H	5CH	5FH	DOH	AOH
		RS-232-C	T.S	WAIT	FREE	GDC
PC9801NS/E 改	2.4576MHz	1251ns	1251ns	1253ns	2033ns	1250ns
386SX-20	150ns NOP	3656ns	18060ns	744ns	2036ns	744ns
PC9801NS/T	1.9968MHz	1581ns	1581ns	1581ns	2629ns	1582ns
386SL-20	151ns NOP	4628ns	993ns	1002ns	2630ns	983ns
PC9801NS 16MHz	2.4576MHz	2507ns	3144ns	3145ns	2499ns	2503ns
386SX-16	189ns NOP	4371ns	2636ns	2637ns	2376ns	1979ns
PC9801NV	1.9968MHz	1120ns	1123ns	1124ns	2504ns	1120ns
V30-16	195ns NOP	4604ns	4602ns	1121ns	2503ns	1120ns
PC98LT V50-8MHz	1.9968MHz	1443ns	1443ns	1443ns	1444ns	1443ns
V50-8	374ns NOP	1441ns	1441ns	1442ns	1442ns	1441ns

〈図 1-C〉 サイリックス CPU に載せ替えたもの

機種名 使用 CPU	8253・クロック NOP 実行速度	I/O アクセス時間(上段=IN, 下段=OUT)				
		30H	5CH	5FH	DOH	AOH
		RS-232-C	T.S	WAIT	FREE	GDC
PC386M 仮想 86	2.4576MHz	1500ns	1598ns	1601ns	3505ns	1503ns
CX486SLC2-16M	89ns NOP	2749ns	1702ns	1703ns	3499ns	1600ns
MELEMM.386Ver5.19						
PC386S 仮想 86	2.4576MHz	1728ns	1620ns	1619ns	2830ns	1615ns
MELEMM.386Ver5.22	115ns NOP	2238ns	892ns	918ns	2664ns	918ns
Cx486DLC25(HSB.EXE CXP)						
PC386S Cx486DLC25	2.4576MHz	1595ns	1411ns	1412ns	2774ns	1409ns
(HSB.EXE CXP)	118ns NOP	2237ns	889ns	913ns	2660ns	912ns
PC9801NS	2.4576MHz	1356ns	1356ns	1356ns	2274ns	1356ns
20MHz+486SLC	149ns NOP	4308ns	918ns	920ns	2278ns	874ns
PC286UX+486SLC	2.4576MHz	1500ns	1502ns	1501ns	2341ns	1501ns
24MHz real mode	149ns NOP	2746ns	1252ns	1252ns	1801ns	1250ns

〈図 1-11〉 ウェイトの入れ方

CPU から周辺 LSI に対して連続アクセスを行う場合、周辺 LSI の動作が完了するまで待ってから、次のアクセスを行わなくてはならない。そのために、8086、V30 を CPU に持つ機種では「NOP」を、80286、80386 を CPU に持つ機種では「JMP __\$+2」を、80486、Pentium を CPU に持つ機種では「OUT __5Fh, AL」等の命令を挿入してタイミングを取る必要がある。

8086 [NOP] 使用(3clock)

LSI		リカバリ・タイム (ns)	WR-WR 5/8	RD-RD 5/8	WR-RD 5/8	RD-WR 5/8
8237	DMAC	400	0 0	0 0	0 1	0 0
8253	タイマ	1000	0 1	0 1	1 2	0 1
8255	PIO	850	0 1	0 1	1 2	0 1
8259	PIC		0 0	0 0	1 1	0 0
8251	モード初期化	6 t_{cy}	3 6			
SIO	非同期モード	8 t_{cy}	4 8			
	同期モード	16 t_{cy}	8 16			
765	FDC		0 0	0 0	0 0	0 0
7220	グラフ	標準 2516	0 3	0 3	0 3	0 3
GDC	2.5MHz	高解像 1710	0 2	0 2	0 2	0 2
7210	GPIO	250	0 0	0 0	0 1	0 0

70116 (V30) [NOP] 使用(3clock)

LSI		リカバリ・タイム (ns)	WR-WR 8/10/16	RD-RD 8/10/16	WR-RD 8/10/16	RD-WR 8/10/16
8237	DMAC	400	0 0 0	0 0 0	1 1 1	0 0 0
8253	タイマ	1000	1 2 3	1 2 3	2 3 4	0 0 2
8255	PIO	850	1 1 2	1 1 2	2 3 4	0 0 1
8259	PIC		0 0 0	0 0 0	0 0 0	0 0 0
8251	モード初期化	6 t_{cy}	6 6 13			
SIO	非同期モード	8 t_{cy}	9 9 19			
	同期モード	16 t_{cy}	20 20 40			
765	FDC		0 0 0	0 0 0	0 0 0	0 0 0
7220	グラフ	標準 2516	4 5 -	4 5 -	5 6 -	3 4 -
	2.5MHz	高解像 1710	2 2 3	2 2 3	3 3 5	1 2 2
GDC	グラフ/テキスト	標準 1260	0 1 -	0 1 -	1 2 -	0 0 -
	5MHz	高解像 855	0 0 0	0 0 0	1 1 1	0 0 0
7201	通信制御	300	0 0 0	0 0 0	1 1 1	0 0 0
7210	GPIO	250	0 0 0	0 0 0	1 1 1	0 0 0

* PC9801NL を除く

80286 [JAP \$+2] 使用(7clock)

LSI		リカバリ・タイム (ns)	WR-WR 8/10/12	RD-RD 8/10/12	WR-RD 8/10/12	RD-WR 8/10/12
8237	DMAC	400	1 1 1	0 0 0	1 1 1	0 0 0
8253	タイマ	1000	1 1 2	1 1 2	1 1 2	1 1 2
8255	PIO	850	1 1 1	1 1 1	1 1 1	1 1 1
8259	PIC		1 0 0	0 0 0	1 1 1	0 0 0
8251	モード初期化	6 t_{cy}	3 3 3			
SIO	非同期モード	8 t_{cy}	4 4 4			
	同期モード	16 t_{cy}	7 7 7			
765	FDC		0 0 0	0 0 0	0 0 0	0 0 0
7220	グラフ	標準 2516	2 3 3	2 2 2	2 3 3	2 2 2
	2.5MHz	高解像 1710	2 2 2	1 1 1	2 2 2	1 1 1
GDC	グラフ/テキスト	標準 1260	1 1 1	1 1 1	1 1 2	1 1 1
	5MHz	高解像 855	1 1 1	0 0 0	1 1 1	0 0 0
7201	通信制御	300	1 1 1	0 0 0	1 1 1	0 0 0
7210	GPIO	250	1 1 1	0 0 0	1 1 1	0 0 0

- ・ EPSON の機種で、PC286、PC286V、PC286U、PC286L、PC286LE 以降の機種と、NEC の PC98DO+ は、周辺 LSI への連続アクセスのためのリカバリ・タイムをハードウェアで生成するために、プログラムでのタイミングを取る必要はない。
- ・ 80486 以上の CPU を持つ機種では、ポート 5FH をライトすることで一定時間のウェイトを確保することができる。EPSON の機種は約 700ns(最低 500ns)のウェイトがかかる。

〈図 1-11〉 ウェイトの入れ方(つづき)

80386

[JAP \$+2] 使用(7+mclock)

LSI		リカバリ・タイム (ns)	WR-WR 12/16/20	RD-RD 12/16/20	WR-RD 12/16/20	RD-WR 12/16/20	
8237	DMAC	400	1 1 1	0 0 0	1 1 1	0 0 0	* 1=2
8253	タイマ	1000	2 2 2	1 1*1	2 2 2	1 1*1	
8255	PIO	850	2 2 2	1 1 1	2 2 2	1 1 1	
8259	PIC		0 0 0	0 0 0	1 1 1	0 0 0	
8251	モード初期化	6 t_{cy}	6 6 6				
SIO	非同期モード	8 t_{cy}	8 8 8				* 2=3
	同期モード	16 t_{cy}	16 16 16				
765	FDC		0 0 0	0 0 0	0 0 0	0 0 0	
7220	グラフ	標準 2516	- - 4	- - 4	- - 4	- - 4	
	2.5MHz	高解像 1710	3 3 3	2 2*2	3 3 3	2 2*2	
	グラフ/テキスト	標準 1260	- - 2	- - 1	- - 2	- - 1	
	5MHz	高解像 855	1 1 1	0*3*3	1 1 1	0*3*3	* 3=0
GDC	テキスト		1 1 1	2*4*5	1 1 1	0*6*7	* x=0
	通信制御	300	1 1 1	0 0 0	1 1 1	0 0 0	
7210	GPIB	250	1 1 1	0 0 0	1 1 1	0 0 0	

* PC9801FX, PC9801FS, PC9821model S1, S2は除く

* 1 PC9801NS/T, PC9801USは1, それ以外は2

* 2 PC9801NS/T, PC9801USは2, それ以外は3

* 3 PC9801DAは1, それ以外は0

* 4 PC9801DAは1, PC9801RS21, 51, PC9801NS/T, C9801NS/Lは2, それ以外は0

* 5 PC9801DAは1, PC9801T, PC9801NS/Lは2, それ以外は0

* 6 PC9801DAは1, PC9801NS/Tは2, それ以外は0

* 7 PC9801DAは1, PC9801Tは2, それ以外は0

80386 PC9801FX, PC9801FS, PC9821model S1, S2

[JAP \$+2] 使用(7+m clock)

LSI		リカバリ・タイム (ns)	WR-WR 12/20	RD-RD 12/20	WR-RD 12/20	RD-WR 12/20
71037	DMAC	125	0 0	0 0	0 0	0 0
71053	タイマ	200	0 0	0 0	0 0	0 0
71055	PIO	200	0 0	0 0	0 0	0 0
71059	PIC	250	0 0	0 0	0 0	0 0
71051	モード初期化	6 t_{cy}	9 14			
SIO	非同期モード	8 t_{cy}	9 14			
	同期モード	16 t_{cy}	9 14			
72065	FDC		0 0	0 0	0 0	0 0
72020	グラフ	標準 2516	4 5	4 5	4 5	4 5
	2.5MHz	高解像 1710	2 4	2 4	2 4	2 4
GDC	グラフ/テキスト	標準 1260	2 3	2 3	2 3	2 3
	5MHz	高解像 855	1 2	1 2	1 2	1 2
7201	通信制御	300	0 0	0 0	0 0	0 0
7210	GPIB	250	0 0	0 0	0 0	0 0

80486, Pentium

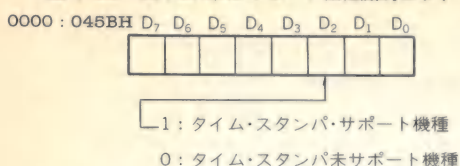
[OUT 5FH, AL] 使用

LSI		リカバリ・タイム (ns)	WR-WR FA/BA/ex	RD-RD FA/BA/ex	WR-RD FA/BA/ex	RD-WR FA/BA/ex
71037	DMAC	125	0 0 0	0 0 0	0 0 0	0 0 0
71053	タイマ	200	0 0 0	0 0 0	0 0 0	0 0 0
71055	PIO	200	0 0 0	0 0 0	0 0 0	0 0 0
71059	PIC	250	0 0 0	0 0 0	0 0 0	0 0 0
71051	モード初期化	6 t_{cy}	6 2 5			
SIO	非同期モード	8 t_{cy}	6 2 5			
	同期モード	16 t_{cy}	6 4 5			
72065	FDC		0 0 0	0 0 0	0 0 0	0 0 0
72020	グラフ	標準 2516	2 - -	2 - -	2 - -	2 - -
	2.5MHz	高解像 1710	1 0 1	1 0 1	1 0 1	1 0 1
GDC	グラフ/テキスト	標準 1260	1 - -	1 - -	1 - -	1 - -
	5MHz	高解像 855	1 0 1	1 0 1	1 0 1	1 0 1
7201	通信制御	300	0 0 1	0 0 0	0 0 1	0 0 0
7210	GPIB	250	0 0 1	0 0 0	0 0 1	0 0 0

FA PC9801FA BA PC9801BA, BX

ex PC9821Ap, As, Ae, Ce, PC9821Ne, PC9801NA, NS/R, NX/C, P, Af

〈図 1-12〉 OUT 5FH,AL サポート機種識別ビット



ジグ機能があり、32 ビットのバスのうち同時に 16 ビットしか使わなくても構いませんので、この機能を使用します。

PC9801/E/F/M 等の初期の PC98 シリーズでは、各 I/O デバイスのアドレス・デコードが、下位 8 ビット分しか行われていませんでした。そのため、8086 自身は 16 ビット (65536 個) のアドレスを持つにもかかわらず、8 ビット (256 個) 分のアドレスしか使用できません。

PC9801VF/VM 以降の機種では一部改められて、本体内部の I/O デバイスが A₁₁, A₁₂ もデコードされるようになりましたので、I/O アドレス空間が 4 倍になったのですが、互換性のためか、拡張された空間はユーザには開放されず、その機種固有の非公開の I/O ポートが割り当てられているようです。

ユーザに開放されている I/O ポート・アドレスは、×nDOH~×nEFH であり、n は 0~7 までです。n=8~F までは NEC のリザーブになっています。7FD9H~7FDFH (奇数) までにはタイマ LSI や、マウス関連に I/O がありますので、下位 8 ビット・デコードのみのボードは使用できません。

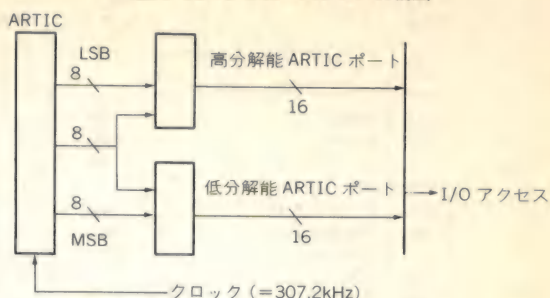
8 ビット・デコードでユーザに解放されている I/O アドレスを使うと、D9H, DBH, DDH, DFH を除いた DOH~EFH の 28 バイトしかありません。他にもこのアドレスを使っている拡張基板は多く、他の基板と併用して使うと、さらに使えるアドレスは減ってしまいます。

厳密には ×nDOH~×nEFH (n=8~F) は NEC のリザーブですから、今後、拡張基板を設計する場合は最低でも 12 ビット・デコードする必要があると思われます。12~16 ビット・デコードすることで、使用できるアドレスが格段に増えます。しかし、フルアドレス・デコードをしても、連続したアドレス空間は最大 32 バイトしか取れません。8 ビット・データ・バスを持つデバイスでは、上位/下位バイトのいずれかししか使えませんので、最大 16 バイトしか連続アドレスが取れません。レジスタをたくさん持つ、高性能な LSI を使用する場合に問題が出そうです。

■ I/O の連続アクセスの制限

CPU から周辺 LSI へ連続したアクセスを行う場合、LSI の処理速度より CPU のアクセスのほうが速くな

〈図 1-13〉 タイム・スタンパの構成



高分解能 ARTIC ポート : ARTIC の下位 16 ビットを読み出すポート

低分解能 ARTIC ポート : ARTIC の上位 16 ビットを読み出すポート

(a) タイム・スタンパの構成

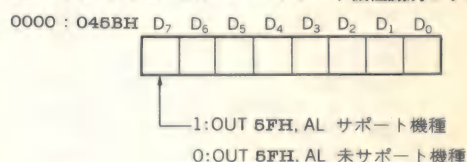
二つの I/O ポートは、設定する時間によって使い分ける

ポート名	分解能	最大値(*1)	アドレス
高分解能 ARTIC ポート	3.26 μ s	213.6ms	005CH (ワード)
低分解能 ARTIC ポート	834.6 μ s	54.7s	005EH (ワード)

(*1) 最上位ビットの変化する周期

(b) ARTIC ポート

〈図 1-14〉 タイム・スタンパ・サポート機種識別ビット



る場合があります。最初に CPU から与えられた命令の処理が終了する前に、次の CPU からの命令が来しまい、LSI が正常な動作をしない場合があります。これを保証するために、周辺 LSI へ連続したアクセスをする場合は、NOP、JMP 命令等でウェイト (リカバリ・タイム) を入れる必要があります (図 1-11)。

8086/V30 系列の CPU では NOP (3 クロック) を、80286/80386 の CPU では JMP \$+2 (7 クロック) (次の命令に JUMP する) を、80486/Pentium では OUT 5FH, AL をいくつか実行することでウェイトを入れます。

CPU に 80486 を持つ機種や PC-H98 シリーズでは、リカバリ・タイムを CPU の実行速度に依存せずに作るために、OUT 5FH, AL という命令を使います。これは I/O ポートの 5FH 番地に書き込みをするだけで、ハードウェア的にウェイトを取る機能で、機種によってかなりばらつきがありますが、最低でも 600 ns

〈図 1-15〉 ディップ・スイッチ SW

参照ポートはプログラム中から読み出すことができるポート

SW ₁	ON	OFF	参照ポート
1	専用高解像度ディスプレイ	標準ディスプレイ	0033H (D3)
2	スーパー・インポーズを使用する	使用しない	0042H (D4)
3	プラズマ・ディスプレイを使用する	使用しない	
4	内蔵 FDD 番号, #3, #4	#1, #2	7FDDH (D0)
5	RS-232-C モード選択 (注 1)		
6	RS-232-C モード選択 (注 1)		7FDDH (D1)
7			
8	拡張グラフィック・モード	基本グラフィック・モード	0042H (D3)

SW ₂	ON	OFF	参照ポート
1			0031H (D0)
2	ターミナル・モード	BASIC モード	0031H (D1)
3	80 字/行	40 字/行	0031H (D2)
4	25 行/画面	20 行/画面	0031H (D3)
5	メモリ・スイッチ保持	メモリ・スイッチ初期化	0031H (D4)
6	内蔵 HDD を切り離す	内蔵 HDD を使用する	0031H (D5)
7	FD モータ制御あり (注 2)	なし	0031H (D6)
8	GDC5MHz モード	2.5MHz	0031H (D7)

SW ₃	ON	OFF	参照ポート
1	内蔵 FD 固定モード	内蔵 FD プログラム・モード	00BEH (D2)
2	内蔵 FD・640K モード	内蔵 FD・1MB モード	00BEH (D3)
3	内蔵 HD・DMA・#1/#2 (注 3) #0		7FDBH (D6)
4			
5	(注 4)		0042H (D1)/7FDDH (D2)
6	RAM512K	RAM640K	
7	メモリ・アクセス 0WAIT (注 5)	1WAIT	
8	動作 CPU・286/386/486	V30	

(注 1) RS-232-C モード選択

5 6 同期モード
ON ON BCI 同期
ON OFF ST2 同期
OFF ON 同期刻時機構
OFF OFF 調歩同期

(注 2) PC9801LV/CV/UV のみ

(注 3) PC98RL/PC9801T/DX/DS/DA/CS/FX/FS/FA/PC9821Ae/As/Ap/Af のみ

(注 4) 機種によって機能が変わる

(注 5) NEC では未定義

- スーパー・インポーズ機能は、PC9801/E/F/M 以外の機種でデジタル RGB 出力を持つ機種のみで使用可能 (オプション)。RGB コネクタの「DOT CLOCK」が入力端子になる。
- プラズマ・ディスプレイは、PC9801/E/F/M 以外の機種でデジタル RGB を持つ機種のみで使用可能 (オプション)。CLOCK とのスキューを保証した RGB 信号を出力する。

程度のリカバリ・タイムが得られます。この機能を持つ機種を判別するには、システム共通領域の 0000:045BH・D7 が“1”であれば、サポートされます (図 1-12)。

また、機種によってはハードウェアで自動的にある程度のリカバリ・タイムを取ってくれるものもありますが、不十分な場合もありますし、他機種での動作保証のためにも、指定された方法でリカバリ・タイムを

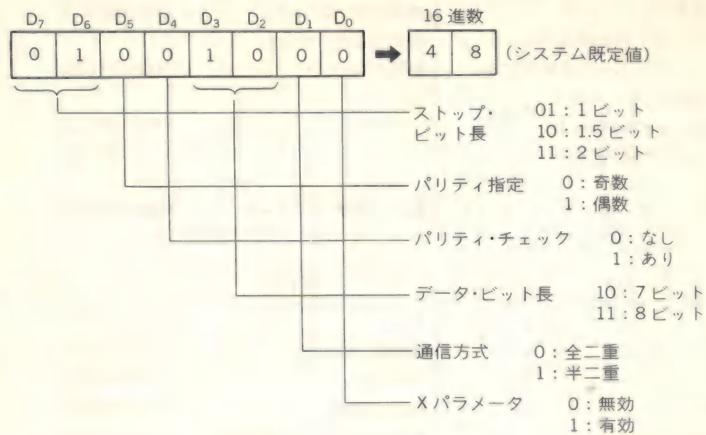
取るほうが良いと思われます (p.10 コラム参照)。

◆ タイム・スタンパ

I/O リカバリ・タイムを保証する手段として、タイム・スタンパを搭載している機種もあります。タイム・スタンパは、クロック 307.2 kHz で動作する 24 ビット・バイナリ・カウンタで、CPU 等に依存せずに動作します。カウンタに対して、初期設定や数値設定

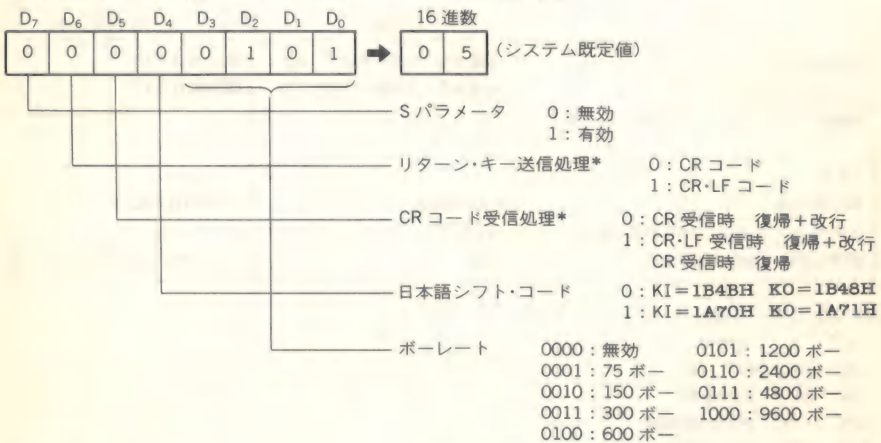
〈図 1-16〉 メモリ・スイッチ

すべてのビットが N₈₈BASIC と MS-DOS で機能する



ビット 0, 1, 2, 3 は N₈₈-BASIC と MS-DOS で機能する

ビット 4, 5, 6, 7 は N₈₈-BASIC のターミナル・モードでのみ機能する



*: CR=0DH, LF=0AH

はできず、フリーランしているのでプログラム中で何度か読み出して、必要な時間が経過したか調べる必要があります。

タイム・スタンパの構成を図 1-13 に示します。

タイム・スタンパは PC-H98 や、CPU に 80486 を採用した機種に搭載されているようで、サポート機種の判別は、図 1-14 のシステム共通領域で識別します。

◆ ディップ・スイッチ

ディップ・スイッチは、機種によってその意味が違っていたり、ハードウェアのスイッチではなくメニュー

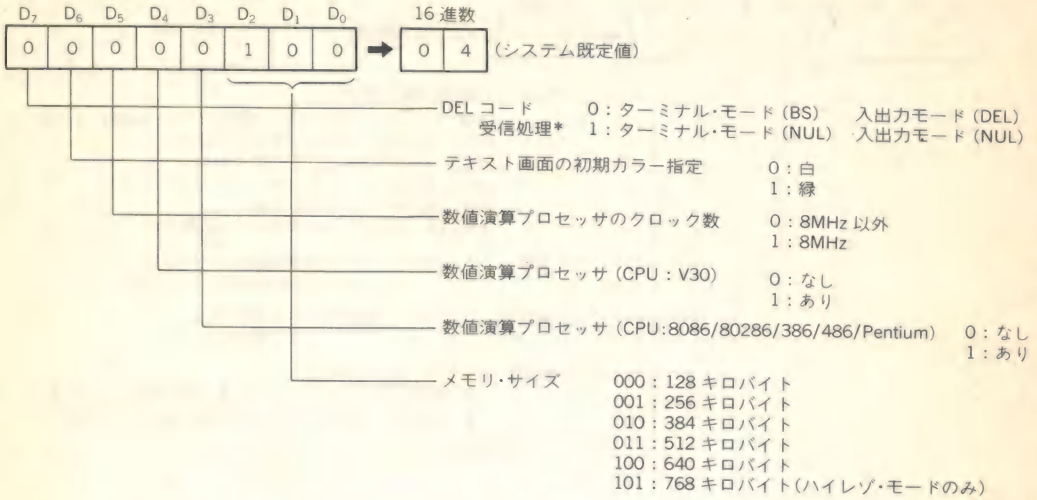
・プログラムから設定する機種もあります。メニュー・プログラムは、「HELP キー」を押しながらリセットすれば起動します。

図 1-15 の表は一般的なディップ・スイッチ表で、一部意味が違う機種もあります。一部または全部がソフトウェア・ディップ・スイッチになっているものや、スイッチの数、順番が違うものもあるので、詳細はそれぞれの機種のマニュアルを参照する必要があります。

図 1-15 にディップ・スイッチ SW の割り付けを示します。

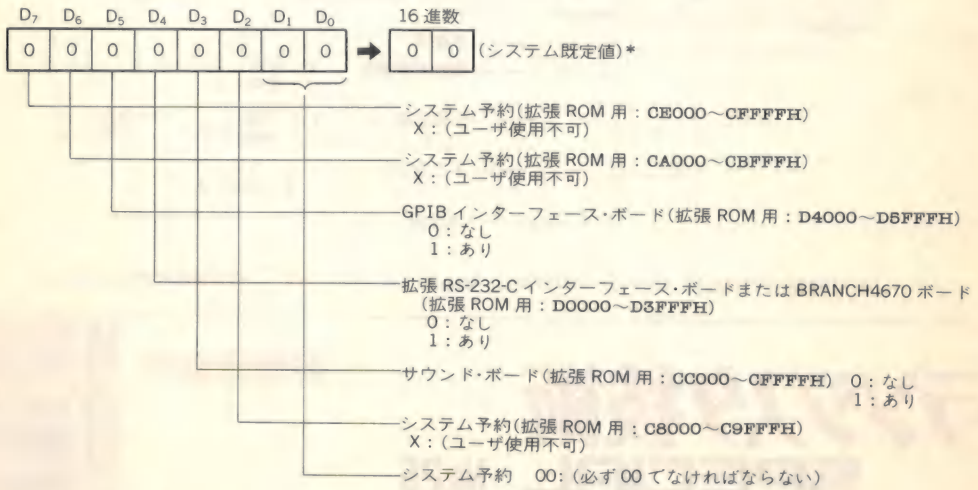
〈図 1-16〉 メモリ・スイッチ(つづき)

ビット 0, 1, 2, 3, 4, 6 は N₈₈ BASIC と MS-DOS で機能する
 ビット 7 は N₈₈ BASIC のみで機能する
 ビット 5 は MS-DOS のみで機能する



*BS=08H, NUL=00H, DEL=7FH または FFH

ビット0, 1, 2, 4, 6, 7はN₈₈BASICとMS-DOSで機能する
ビット3および5はN₈₈BASICのみで機能する



*:サウンド・ボード内蔵機種の場合は 08H

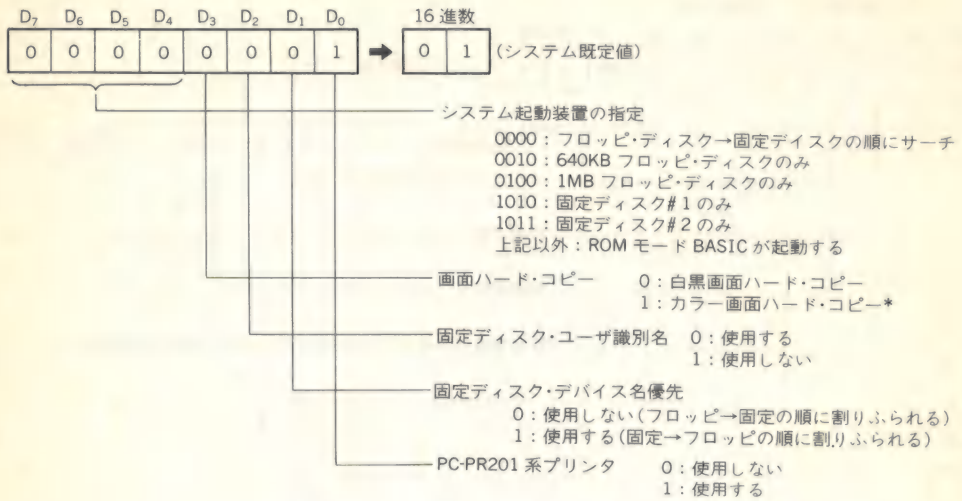
◆ メモリ・スイッチ

メモリ・スイッチは、各種システムの設定値を不揮揮メモリで保存します。RS-232-Cパラメータ、メモリ・サイズ、起動ドライブ指定等の機能があります。

メモリ・スイッチは通常のメモリ空間にあるために、
不用意に書き換えないように、ライト・プロテクトを
かけることができます。不揮発メモリへの書き込み許
可・禁止は I/O アドレス (6AH) のモード・レジスタで
行います (図 1-16)。

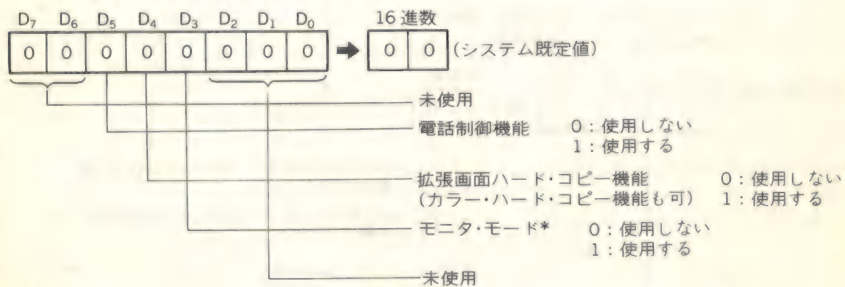
〈図 1-16〉 メモリ・スイッチ(つづき)

ビット 0, 1, 4, 5, 6, 7 は N₈₈BASIC と MS-DOS で機能する
 ビット 2, 3 は N₈₈ BASIC のみで機能する



*: PC-PR201V 系カラー・プリンタが接続され、SW₆のビット 4=1 のときのみ有効

すべてのビットが N₈₈ BASIC のみで機能する



*: PC9801VF2/VM0, 2, 4/UV21/VM21/VX0, 2, 4/UV21では、モニタ・モード拡張機能使用の有無となる

トランジスタ技術 SPECIAL No.25

好評発売中

特集 最新マイコン・メモリ・システム設計法

DRAM,SRAMの動作からデュアルポートRAM,FIFOの活用まで

目次

EPROM活用の基礎技術
 SRAM活用の基礎技術
 疑似SRAM活用の基礎技術
 DRAM活用の基礎技術
 デュアルポートRAM活用の基礎技術

FIFOメモリ活用の基礎技術
 タイミング計算の方法
 8ビットMPUのためのメモリ・システム
 16ビットMPUのためのメモリ・システム
 32ビットMPUのためのメモリ・システム



B5判 160頁
 定価1,540円(税込)

送料 310円

CQ出版社

PC9801 シリーズの割り込み

8086 系の CPU では、リセットを除く割り込みのすべてがベクタ(割り込み先アドレス)が、メモリ空間の先頭の 0000:0000H~03FFH までの 1K バイトの領域に割り当てられています。

ベクタは全部で 256 個あり、CPU の内部処理割り込みやハードウェア割り込み、ソフトウェア割り込み等が割り当てられています。

割り込みベクター一覧を図 1-17 に示します。

一つのベクタは 4 バイトあり、セグメント部 2 バイトとオフセット部 2 バイトで成っています。割り込みがなかった場合は、このアドレスへ FAR JUMP します。

例えば、割り込み 3 番が発生すると、 $3 \times 4 = 12$ で、000CH-000DH のデータをオフセットに、000EH-000FH のデータをセグメントに入れて、FAR JUMP します。

このデータは、小さい番地のアドレス(000CH)に

〈図 1-17〉 割り込みベクタ

ベクタ・アドレス	ベクタ番号	用 途	ベクタ・アドレス	ベクタ番号	用 途
0-3	0	除算エラー	40-43	10	プリンタ/NDP(注 2)
4-7	1	シングル・ステップ	44-47	11	拡張バス INT ₃ (HD)
8-B	2	NMI	48-4B	12	拡張バス INT ₄₁ (640KBFD)
C-F	3	INT ₃	4C-4F	13	拡張バス INT ₄₂ (1MBFD)
10-13	4	オーバ・フロー	50-53	14	拡張バス INT ₅ (サウンド)
14-17	5	コピー(COPY)キー	54-57	15	拡張バス INT ₆ (マウス)(注 3)
18-1B	6	STOP キー	58-5B	16	NDP/プリンタ(注 4)
1C-1F	7	インターバル・タイマ	5C-5F	17	ノイズ(システム予約)(注 6)
20-23	8	タイマ	60-63	18	KB/CRT BIOS
24-27	9	キーボード	64-67	19	RS-232-C BIOS
28-2B	A	CRTV(V-SYNC)	68-6B	1A	プリンタ BIOS
2C-2F	B	拡張バス INT ₀	6C-6F	1B	DISK BIOS
30-33	C	RS-232-C(ch0)	70-73	1C	カレンダー BIOS
34-37	D	拡張バス INT ₁	74-77	1D	システム予約
38-3B	E	拡張バス INT ₂ (CMT)(注 1)	78-7B	1E	N ₈₈ BASIC
3C-3F	F	システム予約	7C-7F	1F	システム予約
			80~FF	20~3F	システム予約
			100~1FF	40~7F	ユーザ用

使用されていないベクタには、デミー処理のアドレスが設定されている。

ベクタ番号 8~17 はハードウェア割り込みに使用。

注 1: PC9801 のみ INT₂は CMT に使用されている。

ハイレゾ・モード時、マウスは INT₅に固定されている。

注 2: 8086, 70116CPU 動作時プリンタ, 80286, 386, 486, Pentium 動作時 NDP。

注 3: ノーマル・モード時、マウスの割り込みはストラップ・スイッチにより INT₀~INT₆に変更可能。

ハイレゾ・モード時、マウスは INT₅に固定されている。

注 4: ノーマル・モードでは 8086, 70116CPU 動作時 NDP, 80286, 386, 486, Pentium 動作時 NC。

ハイレゾ・モード時プリンタ。

注 5: ハイレゾ・モード時のみ GRAPH BIOS。

注 6: PC9821Ap, As, Ae, Af, Ne はシステム・タイマに使用。

〈リスト 1-1〉 割り込みベクタのリスト

```

0000:0000 54 53 F5 09 36 09 80 FD F0 08 80 FD 36 09 80 FD
0000:0010 36 09 80 FD F2 77 60 00 2D 77 60 00 36 09 80 FD
0000:0020 03 06 80 FD 17 09 73 1B 37 09 80 FD 37 09 80 FD
0000:0030 33 18 80 FD 1E 00 00 D4 37 09 80 FD 37 09 80 FD

```


は下位バイトが、大きい番地のアドレス(000DH)には上位バイトが入っています。

割り込みベクタのダンプ・リストをリスト 1-1 に示します。

◆ 割り込みの種類

8086 の割り込みには、CPU 自身が発生する割り込みと、ハードウェアによる割り込みと、ソフトウェアによる割り込みがあります。

CPU 自身が発生する割り込みには、除算エラー、シングル・ステップ、INTO 命令等があり、CPU により割り込みの使用方法が決まっています。ハードウェア割り込みは、CPU 外部から割り込み要求(例えばタイマ LSI)があった場合に処理される割り込みです。ソフトウェア割り込みは、ユーザがプログラムの中から任意に発生させられる割り込みで、サブルーチ的な感覚で使用できます。

◆ 内部処理割り込み (CPU 自身が発生する割り込み)

CPU により使用方法が決められている割り込みには以下の 4 種類があります。

- ① 割り込み番号 0：除算のときに 0 で割り算を行った場合に発生
- ② 割り込み番号 1：シングル・ステップ動作をしているときに発生
- ③ 割り込み番号 3：ブレーク・ポイントのセットに使用
- ④ 割り込み番号 4：INTO 命令のときに発生

シングル・ステップ割り込みは、CPU が 1 命令実行するたびに割り込みがかかります。コントロール・フラグの TF：トラップ・フラグ(D8)をセットすることで、シングル・ステップ動作を開始します。プログラムのデバッグ等に使用するものです。

INTO 命令割り込みは、オーバフロー・フラグがセットされているときに、INTO 命令を実行させるとかかる割り込みです。オーバフロー・フラグがセットされていなければ、INTO 命令が実行されても割り込みはかかりません。

◆ ハードウェア割り込み

CPU 外部からハードウェアで要求できる割り込みには、NMI 割り込みと INT 割り込みがあります。NMI 割り込みはプログラムからマスクできない割り込みで、PC98 シリーズではメモリのパリティ・チェックに使用されています。

割り込み番号 2：NMI 割り込みに使用

INT 割り込みは、プログラムからマスクできる割り込みで、PC98 シリーズでは、割り込みコントロー

ラ(μ PD8259A)を 2 個使用して、全部で割り込み数を 14 個に拡張して使用しています。PC98 シリーズで割り込み処理の中心になる部分です。

割り込み番号 8~14： μ PD8259A(マスタ)により
割り込み

割り込み番号 16~21： μ PD8259A(スレーブ)により
割り込み

◆ ソフトウェア割り込み

ソフトウェア割り込みは、プログラムの中から任意にかけられる割り込みで、

INT $\times \times$ ($\times \times$ は割り込み番号)

という命令で、割り込み実行されます。感覚的にはサブルーチンと同様ですが、セグメントを越えて簡単に呼び出せるために、BIOS の呼び出し等に多用されています。

使用に際しては、割り込み番号 00H~1FH まではシステムが、ハードウェア割り込みや BIOS 呼び出しに使用しています。20H~3FH まではシステム予約とされ、ユーザが自由に使用できる割り込みは 40H~7FH までとされています。

ソフトウェア割り込みの使用状況は、立ち上げるシステムによって変わります。N₈₈BASIC 等では、BAISC の処理等のために、80H~FOH までを使用しています。MS-DOS では、20H~3FH までを MS-DOS を使用し、さらに、常駐する FEP 等がその他の割り込みを勝手に使用します。

◆ 拡張スロットの割り込み制御信号

拡張スロットで使用できる割り込み制御信号には、拡張メモリ用パリティ・チェック用の NMI 割り込みと、INT 割り込みを μ PD8259A で拡張した割り込みのうち 8 種類を使用できます。そのうち、通常は、FDD、HDD、マウス等で 4 種類使いますが、後はユーザが自由に使えるものも多くあります。また、オプション・ボードでも、図 1-18 のようにこれらの割り込みを使用します。

INT₃は一般的にはハード・ディスク・ドライブ・インターフェースで使用されます。PC9801-27(SASI)も、PC9801-55(SCSI)も標準では、この割り込みを使用します。内蔵の IDE 用インターフェースでも INT₃を使用しているようです。

割り込みを INT₃以外に変更できるものもあります。例えば、SASI と SCSI の二つの HDD ドライブ・インターフェースをつなぐ場合、片方の割り込みを INT₃以外で使用する両方使えます。

INT₄₁/INT₄₂は、フロッピー・ディスク・ドライブで使用します。INT₄₁は 640KB 用、INT₄₂は 1MB 用です。内蔵の 640KB/1MB 両用インターフェースで、

〈図 1-18〉 オプション・ボード割り込みレベル使用状況

オプション・ボード		割り込みレベル	INT ₀	INT ₁	INT ₂	INT ₃	INT ₄	INT ₄₁	INT ₄₂	INT ₅	INT ₆
PC9801-05	ODA I/F				◎						
PC9801-08/09	640KB FD I/F						◎				
PC9801-03/13	CMT I/F			◎							
PC9801-14	ミュージック I/F		○				○	○	○	○	◎
PC9801-15	1MB FD I/F							◎			
PC9801-16	68000 ボード										◎
PC9801-26/K	サウンド I/F		○				○	○	◎	○	
PC9801-07/27	HD I/F					◎					
PC9801-06/19/29/K/N	GPIO I/F		○				○	○	◎	○	
PC9801-36	CGMT I/F		○		◎					○	
PC9801-37	ファクシミリ・ボード		◎	○	○					○	
PC9801-50/55/U/L/92	SCSI I/F ボード		○	○	○	◎				○	○
PC9801-59/81	高速回線アダプタ		○				◎	◎	○	○	
PC9861/K	CH2		◎	○	○	○					
RS-232-C 拡張 I/F	CH3		○				○	○	◎	○	
PC9864/U	ネットワーク I/F		◎				○	○	○	○	
PC9862/9866	通信制御アダプタ		◎				○	○	○	○	
PC9871/K	マウス I/F		○	○	○	○	○	○	○	○	◎
本体内蔵マウス I/F			○	○	○	○	○	○	○	○	◎
PC9873	タッチ・スクリーン		◎	○						○	○
PC9801U-03/UV2 内蔵	サウンド I/F									◎	○
PC98XL-02	ImPP ボード		○	◎						○	
PC98XL ² -04	B4680 I/F ボード		○				○	○	◎	○	
(PS98-144-XXX)	PC-UX ボード		○				○	○	○	○	◎
PC9801-77/78	B4680 I/F(NIB)		◎	○	○					○	○
PC9801-83/84	B4680 I/F(NIB)		○	◎	○	○	○	○	○	○	○
PC9801-88	R8100 インターフェース・ボード		○	◎	○	○	○	○	○	○	○
PC9866L	通信制御アダプタ		◎	○			○	○	○		
PC9867/9868	B4680 I/F(IOP)		○				○	○	◎	○	
PC9801-82	GPIO ボード		○	○	○					○	○

◎：工場出荷時設定 ○：変更可能レベル

640KB/1MB 自動切り替えモードで使用する場合は INT₄₁を使用しますので、普通両用ドライブを搭載している機種では、INT₄₂は拡張スロットには出ていません。

INT₅は、サウンド・ボード(FM 音源ボード)で使用されます。サウンド・ボード内蔵の機種では INT₅に

固定されているものもあります。

INT₆は一般的にはマウス・インターフェースで使用されます。INT₆以外に変更できる機種もありますが、98NOTE や最近の機種では固定されているものもあります。

トランジスタ技術

SPECIAL No.29

特集 マイコン独習Z80完全マニュアル

手作りの原点から実用ソフトの作成まで

好評発売中

B5判 160頁
定価1,540円(税込)

CQ出版社 〒170 東京都豊島区巣鴨1-14-2 営業部 ☎(03)5395-2141 振替

システム共通領域

0000 : 0400H~05FFH にシステム共通領域というエリアがあり、システムの情報、設定の識別や BIOS 等のワーク・エリアになっています。この領域を調べることで、機種依存性のある機能の有無を調べ、使用するか否かを決定できます。

この領域は参照するだけで、書き込みを行ってはいけません。

以下にシステム共通領域の機能を示します。

0400H

D ₀	
D ₁	
D ₂	
D ₃	CPU 0=V30/V50 1=V33
D ₄	
D ₅	1=レジューム有効(E)
D ₆	
D ₇	RAM ドライブ 1=搭載

0401H 拡張 RAM サイズ(128K 単位) (E)

0404H

{ リアル・モード復帰時の SP

0405H

0406H

{ リアル・モード復帰時の SS

0407H

0458H

D ₀	CPU 0=386DX 1=386SSX
D ₁	
D ₂	
D ₃	CPU 1=486SX(PC-H98 のみ)
D ₄	
D ₅	
D ₆	0=その他 1=マルチ・メディア(PC98GS)
D ₇	0=9801-BUS 1=NESA-BUS

045AH

D ₀	0		0	
D ₁	0		0	
D ₂	0		0	
D ₃	0		0	
D ₄	0	486SX-16	0	386SX-10
D ₅	1		0	
D ₆	0		1	
D ₇	0		0	

045BH

D ₀	
D ₁	

D ₂	タイム・スタンパ 1=搭載
D ₃	
D ₄	
D ₅	
D ₆	
D ₇	OUT 5FH,AL ウェイト 1=搭載

045CH

D ₀	
D ₁	
D ₂	
D ₃	
D ₄	
D ₅	CRT 垂直同期周波数* 1
D ₆	1=256 色表示可能機種
D ₇	1=486SX(PC-H98 以外)

* 1 D₅=ノーマル・モード時 0=24.83kHz, 1=31.47kHz

ハイレゾ・モード時 0=32.84kHz, 1=50.0kHz

0480H CPU 種別フラグ

D ₀	CPU	0	V30/286	1	286	1	386/486/Pentium
D ₁		0	(XA)	0		0	
D ₂							
D ₃							
D ₄							FD モータ制御 1=可能(1MB/1.44MB)
D ₅							
D ₆							
D ₇							

0481H システム環境フラグ

D ₀	
D ₁	
D ₂	
D ₃	キーボード* 1
D ₄	
D ₅	ソフトウェア EMS 1=使用可能
D ₆	キーボード* 1
D ₇	

* 1

D ₃	D ₆	
0	0	旧式
0	1	RA,RS,RX,ES,EX,DA,DS,DX,CS,FA,FS,FX,US,BX,BA,Ap,As,Ae,Ce,Af,RL,GS
1	0	ノート/ラップトップ DIP SW ₂₋₇ ON
1	1	ノート/ラップトップ DIP SW ₂₋₇ OFF

0482H SCSI HDD 接続フラグ (E)

0484H HDD 環境フラグ

D ₀	CPU 1=386SX/SL
D ₁	0 386-10 0
D ₂	CPU 0 386-10/12/16 0 286-12 386-20
D ₃	0 0

D ₄	内蔵 SASI HDD 1=あり (注 1)			
D ₅	内蔵 SCSI HDD 1=あり (注 1)			
D ₆	DMA チャンネル	0	CH 0	1 CH 1 (注 1)
D ₇		0		0

(注 1) EPSON PC386GS/GE 以降で DMA 変更可能な機種

0486H

{ CPU のバージョン (EPSON PC486GR/GF 以降)

0487H

0488H RAM ドライブ接続フラグ

D ₀	1MB	# 0	1=接続あり
D ₁		# 1	
D ₂		# 2	
D ₃		# 3	
D ₄	640KB	# 0	
D ₅		# 1	
D ₆		# 2	
D ₇		# 3	

0492H 1MB/640KB モード・チェンジ

D ₀	1MB	# 0	モード変更されたとき「1」になる
D ₁		# 1	キャリプレートを行うと「0」になる
D ₂		# 2	
D ₃		# 3	
D ₄	640KB	# 0	
D ₅		# 1	
D ₆		# 2	
D ₇		# 3	

0493H 1MB インターフェースのアクセス・モード

D ₀	サーフェイス	# 0	0=片面, 1=両面
D ₁		# 1	
D ₂		# 2	
D ₃		# 3	
D ₄	トラック	# 0	0=40トラック, 1=80トラック
D ₅		# 1	
D ₆		# 2	
D ₇		# 3	

0494H 1MB ドライブ接続 (E)

D ₀			
D ₁			
D ₂			
D ₃			
D ₄	1MB	# 0	1=接続あり
D ₅		# 1	
D ₆		# 2	
D ₇		# 3	

04ACH

{ 拡張ROMイニシャルのポインタ・オフセット・アドレス

04ADH

04AEH

{ 拡張ROMイニシャルのポインタ・セグメント・アドレス

04AFH

04BOH

{ 拡張デバイスのセグメント値の上位8ビット(16個分)

04BFH

04DOH

{ 拡張 ROM ID (16 個分)

04DFH

0500H システム・フラグ 1

D ₀	システム既定ビット 1			
D ₁	0=640KB I/F 1=1MB I/F			
D ₂	640KB AI 検出 1=許可			
D ₃	VSYNC (?)			
D ₄	拡張 RAM 1=あり (?)			
D ₅	キー・バッファ・オーバフロー時のブザー 0=鳴らす			
D ₆	システム既定ビット 2 (DISK BASIC)			
D ₇	ブート種別 0=コールド・ブート 1=ウォーム・ブート			

0501H

D ₀	メモリ・	0	128	1	256	0	384	1	512	0	640
D ₁	サイズ	0	KB	0	KB	1	KB	1	KB	0	KB
D ₂		0		0		0		0		1	
D ₃	0=ノーマル・モード 1=ハイレゾ・モード										
D ₄	0	PC9801, XA	0	PC9801, XA,	1	U2, LT					
D ₅	0		1	U2, LT 以外	1						
D ₆	CPU 0=8086/286 1=V30										
D ₇	システム・クロック 0=5/10MHz 1=8MHz										

0502H

{ キーボード・バッファ (16 文字分)

0521H

0522H

{ キーボード変換テーブル・オフセット・アドレス

0523H

0524H

{ キーボード・バッファ・データ読み込みポインタ

0525H

0526H

{ キーボード・バッファ書き込みポインタ

0527H

0528H キーボード・バッファ・データ・ワード数

0529H KB ERROR (?)

052AH

{

0539H キーボード・キー押下状態ビット・マップ (16 ビット)

053AH 特殊キー押下状態

D ₀	SHIFT	0=押下 1=開放
D ₁	CAPS	
D ₂	カナ	
D ₃	GRPH	

D ₄	CTRL	
D ₅		
D ₆		
D ₇		

053BH 1行中の表示ライン数-1

053CH CRTステータス・フラグ

D ₀	画面モード 0=25行 1=20行
D ₁	画面モード 0=80桁 1=40桁
D ₂	アトリビュート 0=バーチカル・ライン 1=簡易グラフィック
D ₃	漢字CGアクセス・モード 0=コード・アクセス 1=ドット・アクセス
D ₄	
D ₅	
D ₆	
D ₇	CRTタイプ 0=標準CRT 1=高解像CRT

053DH CRT BIOS WORK

053EH

{ CRT CONT. DATA ADDRESS

0541H

0542H

{ CRTV VECTOR

0545H

0546H CR FONT

0547H GDC SCROLL AREA

0548H

{ PRINT VRAM ADDRESS

0549H

054AH

{ LINE 200/400

054BH

054CH グラフィック・ステータス

D ₀	0=標準モード 1=拡張モード
D ₁	1=グラフィック・チャージャあり
D ₂	1=16色ボードあり
D ₃	1=ユーザ定義文字 188文字
D ₄	
D ₅	
D ₆	0=標準CRT 1=高解像CRT
D ₇	0=グラフィック画面表示中 1=グラフィック画面表示停止

054DH ドット・オペレーション・モード

D ₀	描画	0	Replace	1	Complement	0	Clear	1	Set
D ₁	モード	0		0		1		1	
D ₂	GDCクロック 0=2.5MHz 1=5MHz								
D ₃	0=フラッシュレス 1=フラッシュ								
D ₄									
D ₅	GDC5MHz 0=禁止 1=許可								
D ₆	1=拡張グラフィック描画機能あり								
D ₇									

054EH

{ GDC LINE STYLE / FONT

0555H

0556H

{ RS-232-C 受信バッファ・オフセット・アドレス

0557H

0558H

{ RS-232-C 受信バッファ・セグメント・アドレス

0559H

055AH ODA PRINTER SHIFT

055BH データ・シフト・ステータス

D ₀					
D ₁					
D ₂	CH	# 2	0 = SI	CH # 0 内蔵 RS-232-C	
D ₃		# 1	1 = SO	CH # 1 増設 RS-232-C 2	
D ₄		# 0		CH # 2 増設 RS-232-C 3	
D ₅	CH	# 2	0=SI/SO 無効 1=SI/SO 有効		
D ₆		# 1			
D ₇		# 0			

055CH 1MB インターフェース FDD 接続

D ₀	#0	1MB・I/F の接続 1=接続あり
D ₁	#1	
D ₂	#2	
D ₃	#3	(2D-FDD)
D ₄	#0	
D ₅	#1	
D ₆	#2	
D ₇	#3	

055DH HDD/FDD インターフェース接続

D ₀	#0	HDD 接続 1=接続あり
D ₁	#1	
D ₂		
D ₃		640KB・I/F の接続 1=接続あり
D ₄	#0	
D ₅	#1	
D ₆	#2	
D ₇	#3	

055EH 1MB インターフェース FDD インタラプト

D ₀	#0	1MB・I/F 1=インタラプトあり
D ₁	#1	
D ₂	#2	
D ₃	#3	
D ₄		
D ₅		
D ₆		
D ₇		

055FH HDD/FDD インタラプト

D ₀	#0	HDD インタラプト 1=あり
D ₁	#1	

D ₂	
D ₃	
D ₄ #0	640KB・I/F インタラプト 1=あり
D ₅ #1	
D ₆ #2	
D ₇ #3	

0560H 2D-FDD 0=NON FF=1SIDE EF=2SIDE

0561H 2D-FDD OPERATION MODE

0562H

{ 2D-FDD TIMEOUT COUNTER

0563H

0564H

{ 1MB・I/F, μ PD765 リザルト・ステータス

0583H (4ドライブ分)

0584H システムがブートしたデバイス

D ₀	UA ₀	DA	0FH=2HD FDD(640KB I/F) UA 0=#1
D ₁	UA ₁		09H=2HD FDD(1MB I/F) 1=#2
D ₂	UA ₂		08H=HDD(SASI) 2=#3
D ₃	UA ₃		07H=2DD FDD(640KB I/F) 3=#4
D ₄	DA ₀		01H=2DD FDD(1MB I/F)
D ₅	DA ₁		0AH=HDD(SCSI)
D ₆	DA ₂		
D ₇	DA ₃		

0585H HDD STATUS

0586H

{ ハード・ディスク・コントローラからのエラー情報

0589H

058AH

{ インターバル・タイマ BIOS 用 タイマ値

058BH

058EH

{ GRAPHICS BIOS/LIO WORK

058FH

05C0H ディップ・スイッチ SW₂(?)

D ₀	SW ₂₋₁	0=ON 1=OFF
D ₁	SW ₂₋₂	
D ₂	SW ₂₋₃	
D ₃	SW ₂₋₄	
D ₄	SW ₂₋₅	
D ₅		
D ₆		
D ₇		

05C1H RS-232-C DEL コマンド

D ₀	CH #0	0=何もしない 1=メモリ・スイッチ SW ₃₋₇ の設定による
D ₁	CH #1	
D ₂	CH #2	
D ₃		

05C2H

{ GRAPHICS BIOS POINTER

05C5H

05C6H

{ キー・コード変換テーブル・ベース状態オフセット・

05C7H アドレス (EPSON-PC486GR/GF 以降)

05C8H

{ キー・コード変換テーブル・ベース状態セグメント・

05C9H アドレス (EPSON-PC486GR/GF 以降)

05CAH 640KB インターフェースのアクセス・モード

D ₀	サーフェイス	#0	0=片面 1=両面*
D ₁		#1	
D ₂		#2	
D ₃		#3	
D ₄	トラック	#0	0=40トラック 1=80トラック
D ₅		#1	
D ₆		#2	
D ₇		#3	

05CBH 640KB インターフェース FDD モータ OFF タ
イマ・カウンタ

05CCH

{ 640KB・FDD コマンド・パラメータ・ポインタ・

05CDH オフセット

05CEH

{ 640KB・FDD コマンド・パラメータ・ポインタ・

05CFH セグメント

05DOH 640KB インターフェース FDC リザルト・ステータス ST₀

05D1H 640KB インターフェース FDC リザルト・ステータス ST₁

05D2H 640KB インターフェース FDC リザルト・ステータス ST₂

05D3H 640KB インターフェース FDC リザルト・ステータス C

05D4H 640KB インターフェース FDC リザルト・ステータス H

05D5H 640KB インターフェース FDC リザルト・ステータス R

05D6H 640KB インターフェース FDC リザルト・ステータス N

05D7H 5 インチ FDD モータ OFF タイム・カウンタ

05D8H シーク・リキャリプレート ST₀# 0~3

{ シーク・リキャリプレート PCN₀# 0~3

05DFH

05F8H

{ 1MB・FDD ディスク・パラメータ・ポインタ・

05F9H オフセット

05FAH

{ 1MB・FDD ディスク・パラメータ・ポインタ・

05FBH セグメント

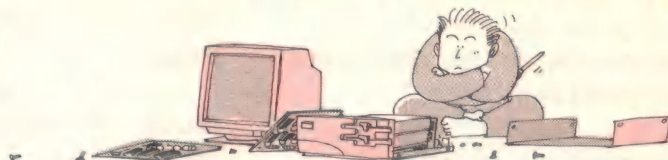
05FEH

{ BASIC LIO DATA セグメント値

05FFH

2

ハードウェアの詳細



割り込みコントローラ

▶使用 LSI

μ PD8259A 相当

▶I/O アドレス

00H, 02H(マスタ)

08H, 0AH(スレーブ)

▶初期設定命令

ICW₁=11H(マスタ/スレーブ)

ICW₂=08H(マスタ), ICW₂=10H(スレーブ)

ICW₃=80H(マスタ), ICW₃=07H(スレーブ)

ICW₄=1DH(マスタ), ICW₄=09H(スレーブ)

■ 割り込みコントローラのハードウェア

PC98 シリーズでは、ハードウェア割り込み制御に 2 個の μ PD8259A 相当品(以下 8259 と略す)を使用しています。8259 は 1 個で 8 本の割り込みを処理でき、さらに直列に接続(マスタとスレーブ)することで、計 14(15)本の割り込みが処理できるようになります(図 2-1)。

マスタ側の 8259 が、スレーブ用の割り込み受け付

けのために 1 チャンネル取られて 7 個、スレーブ側の IRQ₇は未使用になっているので 7 個あります。接続は図 2-2 のようになっています。

割り込みの使用状況は、本体内部で 6 本、拡張バスに 8 本です。PC9821Ap/As/Ae/Af/Ne では IR₁₅(スレーブの IRQ₇)をシステム・タイマに使用しています。

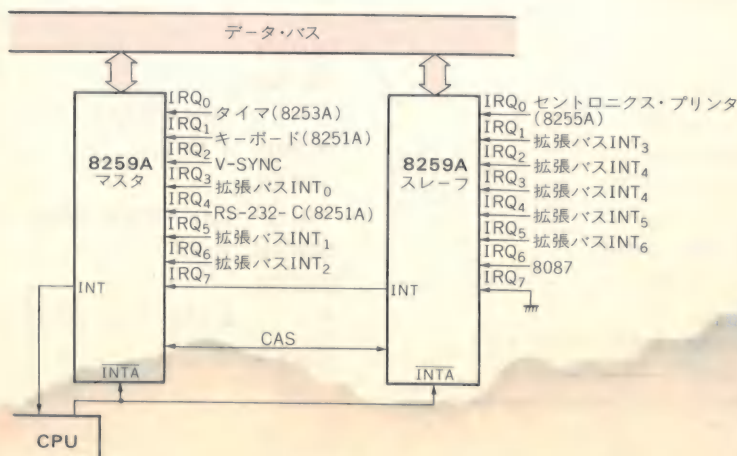
本体内部には、タイマ、キーボード、CRTV、RS-232-C、プリンタ、NDP 等に使用されています。拡張バス用に割り当てられた中から、フロッピー・ディスク・インターフェース(2DD/2HD)や、ハード・ディスク・インターフェース、マウス・インターフェースで計 4 本使用します。残りの割り込みは、ユーザが独自の拡張ボードで使用できます。

また、フロッピー・ディスク・インターフェースの割り込みレベルは固定になっていますが、ハード・ディスク・インターフェースと、マウス・インターフェースは、機種によっては割り込みレベルを変更することができます(図 2-3)。

CPU と 8259 との割り込み制御線は、割り込み要求(INTR)と、割り込み許可(INTA)の 2 本だけです。これで、14(15)個もの割り込みを制御できるのは、8259 が割り込みを受け付けると、割り込み要求番号に対応した、割り込みペクタ番号を CPU に対して送

〈図 2-2〉

2 個の割り込みコントローラの接続図



〈図 2-1〉 割り込みコントローラの I/O アドレス一覧

デバイス名	命 令	READ/ WRITE	I/O ポート ・アドレス	デ ー タ								備 考
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
マ ス タ	ICW ₁	W	00	0	0	0	1	LTIM	0	S	1	S=0
	ICW ₂	W	02	T ₇	T ₆	T ₅	T ₄	T ₃	0	0	0	T ₇ ~T ₃ =00001
	ICW ₃	W	02	1	0	0	0	0	0	0	0	
	ICW ₄	W	02	0	0	0	SFNM	BUF	1	0	1	BUF =1
	OCW ₁	W	02	M ₇	M ₆	M ₅	M ₄	M ₃	M ₂	M ₁	M ₀	
	OCW ₂	W	00	R	SL	EOL	0	0	L ₂	L ₁	L ₀	
	OCW ₃	W	00	0	ESMM	SMM	0	1	P	RR	RIS	
	ボール・モード	R	00	I	X	X	X	X	W ₂	W ₁	W ₀	
	IRR リード	R	00	IR ₇	IR ₆	IR ₅	IR ₄	IR ₃	IR ₂	IR ₁	IR ₀	
	ISR リード	R	00	IS ₇	IS ₆	IS ₅	IS ₄	IS ₃	IS ₂	IS ₁	IS ₀	
ス レ ー プ	IMR リード	R	02	M ₇	M ₆	M ₅	M ₄	M ₃	M ₂	M ₁	M ₀	
	ICW ₁	W	08	0	0	0	1	LTIM	0	S	1	S=0
	ICW ₂	W	0A	T ₇	T ₆	T ₅	T ₄	T ₃	0	0	0	T ₇ ~T ₃ =00010
	ICW ₃	W	0A	0	0	0	0	0	1	1	1	
	ICW ₄	W	0A	0	0	0	SFNM	BUF	0	0	1	BUF =1
	OCW ₁	W	0A	M ₁₅	M ₁₄	M ₁₃	M ₁₂	M ₁₁	M ₁₀	M ₉	M ₈	
	OCW ₂	W	08	R	SL	EOI	0	0	L ₂	L ₁	L ₀	
	OCW ₃	W	08	0	ESMM	SMM	0	1	P	RR	RIS	
	ボール・モード	R	08	I	X	X	X	X	W ₂	W ₁	W ₀	
	IRR リード	R	08	IR ₁₅	IR ₁₄	IR ₁₃	IR ₁₂	IR ₁₁	IR ₁₀	IR ₉	IR ₈	
	ISR リード	R	08	IS ₁₅	IS ₁₄	IS ₁₃	IS ₁₂	IS ₁₁	IS ₁₀	IS ₉	IS ₈	
	IMR リード	R	0A	M ₁₅	M ₁₄	M ₁₃	M ₁₂	M ₁₁	M ₁₀	M ₉	M ₈	

出すことで、CPU がそれを読み取り、対応する割り込みベクタの内容のアドレスを呼び出すことで実現しています。

8259 が送出する割り込みベクタ番号は、8259 を初期化するときに設定します。つまり、割り込み処理用に割り当てられているベクタ番号 08H~17H は、再初期設定しなせば可変可能です。しかし、変更する意味はありません。

◆ 割り込みコントローラの制御

8259 は、「IMR, IRR, ISR」の三つのレジスタと、初期設定用の四つのイニシャライズ・ワード、コマンド設定用の三つのコマンド・ワードがあり、それらを二つの I/O アドレスからアクセスします。マスタ用 8259 が 00H, 02H 番地で、スレーブ用が 08H, 0AH になります。ICW₁/OCW₂/OCW₃は書き込むときのデータ(ビット 3, 4)の設定で決定されます(図 2-4)。

▶ 割り込みマスク・レジスタ (IMR)

アドレス：02H(マスタ)

0AH(スレーブ)/リード・ライト

割り込みマスク・レジスタ IMR は、コマンド・ワードの割り込みマスク・ワード OCW₁を格納するところ

です。このレジスタのビットが“1”ならば、割り込み要求があっても 8259 は受け付けません。

割り込みのマスクには、他にも CPU の命令の「CLI, STI」があります。これは、割り込みすべてを禁止しますが、IMR では個別に割り込みのマスクを行います。

また、割り込みを発生するデバイス自身にも、割り込みのマスクができるものも多くあります。

▶ 割り込み要求レジスタ (IRR)

アドレス：00H(マスタ)

08H(スレーブ)/リード・OCW₃の RIS=0 コマンド・ワード OCW₃の RIS が“0”のときに有効になります。通常は、IRR を読み出す前に「00H(マスタ)/08H(スレーブ)に 0AH」を書き込みます。

割り込み要求レジスタ IRR は、8 レベルの割り込み入力のうち、現在割り込み要求を出しているレベルを示す情報を持っています。このレジスタのビット(D₀~D₇)は、割り込み要求入力(IR₀~IR₇)に対応しています。

▶ インサービス・レジスタ (ISR)

アドレス：00H(マスタ)

08H(スレーブ)/リード・OCW₃の RIS=1 コマンド・ワード OCW₃の RIS が“1”のときに有

〈図 2-3〉 割り込みレベルとベクタ番号

デバイス名	割り込み 要求信号	レベル	割 り 込 み 名	ベクトル・データ								ベクタ番号
				T ₇	T ₆	T ₅	T ₄	T ₃	T ₂	T ₁	T ₀	
μPD8259A /μPD71059 (マスタ)	IR ₀	0	タイマ	0	0	0	0	1	0	0	0	08
	IR ₁	1	キーボード	0	0	0	0	1	0	0	1	09
	IR ₂	2	CRTV	0	0	0	0	1	0	1	0	0A
	IR ₃	3	拡張バス INT ₀	0	0	0	0	1	0	1	1	0B
	IR ₄	4	RS-232-C	0	0	0	0	1	1	0	0	0C
	IR ₅	5	拡張バス INT ₁	0	0	0	0	1	1	0	1	0D
	IR ₆	6	拡張バス INT ₂	0	0	0	0	1	1	1	0	0E
	IR ₇	7	スレーブ	0	0	0	0	1	1	1	1	0F
μPD8259A /μPD71059 (スレーブ)	IR ₈	8	プリンタ (70116) /NDP (80286/386)	0	0	0	1	0	0	0	0	10
	IR ₉	9	拡張バス INT ₃ (固定ディスク)	0	0	0	1	0	0	0	1	11
	IR ₁₀	10	拡張バス INT ₄₁ (640KB FD)	0	0	0	1	0	0	1	0	12
	IR ₁₁	11	拡張バス INT ₄₂ (1MB FD)	0	0	0	1	0	0	1	1	13
	IR ₁₂	12	拡張バス INT ₅	0	0	0	1	0	1	0	0	14
	IR ₁₃	13	拡張バス INT ₆	0	0	0	1	0	1	0	1	15
	IR ₁₄	14	NDP (70116) /なし (80286/386)	0	0	0	1	0	1	1	0	16
	IR ₁₅	15	(システム・タイマ)	0	0	0	1	0	1	1	1	17

* マスタおよびスレーブの T₇~T₃は、それぞれ独立にプログラムで値をセットする

* IR₈および IR₁₄は CPU によって変化する。また、80287/387 の割り込みは 80286/386 に直結されているため、80287/387 の割り込みは 8259 では制御できない

* マウスの割り込みレベルなどは、ハイレゾ・モード時は INT₂(IR₆)に固定されているが、ノーマル・モード時ではストラップ・スイッチなどにより INT₀~INT₆のいずれかを選択可能(規定値は INT₀)。98NOTE, PC9801BA, BX, PC9821Ap, As, Ae, Af では INT₀(IR₁₅)に固定されている

* IR₁₅は不用意にマスクしないこと

* IR₁₅は PC9821Ap, As, Ae, Af, Ne にてシステム・タイマに使用される。その他の機種では何もせずにリターンする

〈図 2-4〉 I/O アドレスとレジスタ

アドレス	データ		READ/ WRITE	レジスタ/コマンド	備 考
	マスタ/スレーブ	D ₄ D ₅			
00H/08H	1	0	WRITE	ICW ₁	初期設定
02H/0AH	×	×	WRITE	ICW ₂ , ICW ₃ , ICW ₄	
00H/08H	0	0	WRITE	OCW ₂	EOI コマンド IRR/ISR 選択 OCW ₃ で選択 IMR 書き込み IMR 読み込み
00H/08H	0	1	WRITE	OCW ₃	
00H/08H	×	×	READ	IRR/ISR	
02H/0AH	×	×	WRITE	OCW ₁	
02H/0AH	×	×	READ	IMR	
02H/0AH	×	×	READ	IMR	

効になります。通常は、ISR を読み出す前に「00H (マスタ)/08H (スレーブ)に 0BH」を書き込みます。

インサービス・レジスタ ISR は、**現在処理中(サービス)のすべての割り込みレベルを示す情報**を持っています。IRR と同様にビットが“1”ならば、割り込みルーチンがサービス中であることを示します。

◆ 割り込みのしくみ (完全ネスト・モードの場合)

8259 には「IR₀~IR₇(割り込み要求信号入力端子)」があり、他のデバイス(タイマ LSI 等)の割り込み要求出力端子に接続されています。他のデバイスからの割り込みが発生すると、IR₀~₇がアクティブになり、IRR レジスタの割り込みに対応するビット(0~7)が立ちます。

割り込みがかかると、8259 は **CPU に対して割り込み要求信号(INTR)を送る**と、CPU は現在実行中の

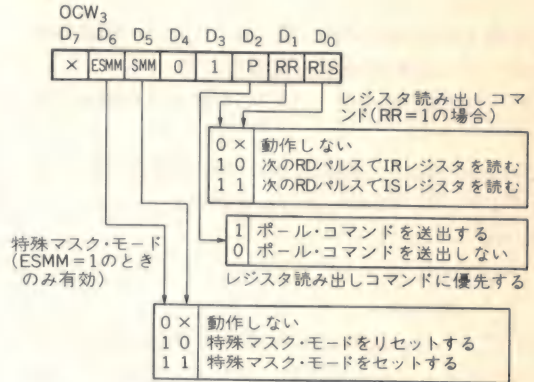
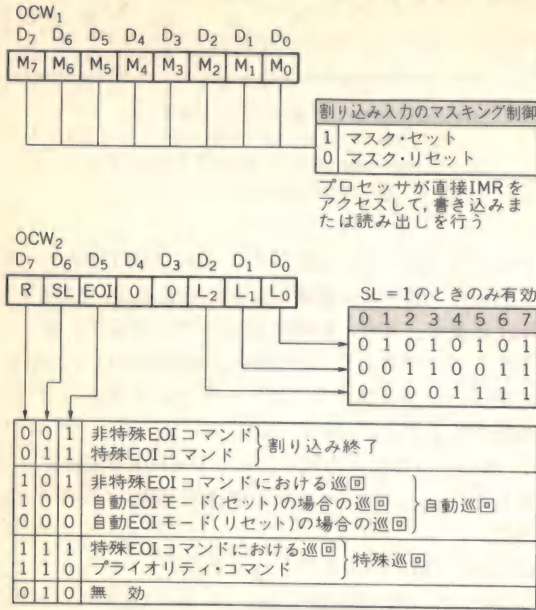
処理を中止させ、割り込み許可信号(INTA)を 8259 へ返します。

8259 は、その要求されている割り込みのうち、**最も優先順位の高い割り込みを ISR レジスタにセットして、あらかじめセットされている 8 ビットの 0 ~255 までの割り込み番号の中の一つを CPU に送ります。**

CPU は、その**割り込み番号に対応する割り込みベクタ**に書いてある**アドレスを読み込み**、そのアドレスへ**制御を移します**。その後、IRET 命令を実行すると、割り込み発生前に処理に戻ります。

割り込みプログラム処理中に、新たに割り込みが入った場合の処理は 2 種類あります。**現在処理中の割り込みの優先順位より、新たに入った割り込みの優先順位が高ければ、いままで行われていた処理を中断して、新しい割り込み処理を行います。**その反対に、現在の割り込みの優先順位より、新たに入った割り込みの優先順

〈図 2-5〉 8259A の動作コマンド・ワード (OCW) のプログラム・フォーマット



位が低ければ、現在の割り込み処理が終了するまで新しい割り込み処理は実行されません。

これらの処理は、8259が独自に管理しています。そのために、割り込み処理の終了を8259に知らせる必要があります。割り込みの終了を知らせないと、ISRレジスタの割り込みレベルに対応するビットはいつまでもクリアされず、その割り込みよりも優先度の低い割り込み要求は待たされたままになってしまいます。

このため、通常は割り込み処理の終了にEOI命令を8259に送らなければなりません。8259がEOIを受けると、そのレベルの割り込みは終了し、他の割り込みを受けられるようになります。

EOI命令はコマンド・ワードの実行で送ることができます。

特殊完全ネスト・モード

マスタにかかった割り込み処理は、マスタが直接割り込み管理できますので問題ありませんが、スレーブにかかった割り込みは、マスタから見るとすべて(7~8個)が一つの割り込み入力に現れます。このため、マスタ側ではスレーブでの割り込みの優先順位が管理できません。

特殊完全ネスト・モードは、スレーブからきた割り込み信号はすべて処理され、その優先順位の処理はスレーブに任せることができます。この場合の割り込み終了の処理は、EOIをマスタとスレーブの両方に送らなくてはなりません。ただし、スレーブが複数の割り込みを受けている場合、ISRレジスタを監視して、ス

レーブのすべての割り込みが終了するのを待ってマスタにEOI命令をかける必要があります。

割り込みコントローラのコマンド実行

コマンド・ワードには三つあります(図2-5)。これを指定されたI/Oアドレス・ポートに書き込めば実行されます。IMRへの書き込み、IRR/ISRの読み出し選択等を設定します。

OCW₂/OCW₃は同じアドレスに書き込みますが、この識別は「D₃、D₄」で区別します。

OCW₁

アドレス: 02H(マスタ),

0AH(スレーブ)/ライト

IMRへの書き込みをします。IRRをマスクして該当する割り込み入力からの割り込み要求を禁止します。また、完全ネスト・モードではISRもマスクします。

0=該当割り込みレベル・マスク

1=該当割り込みレベル許可

OCW₂

アドレス: 00H(マスタ),

08H(スレーブ)/ライト

このワードは、割り込み処理の終了(EOI)を宣言するコマンドや、割り込み順位を変更するコマンドを設定します。

D₇: R……このビットは、割り込み要求の優先順位を変更(回転)するためのビットで、R=1のときに変更します。割り込みを終了するたびに、優先順位を変更することで特定の割り込みに処理を集中させないようにできます。

D₆: SL……このビットは、L₀～L₂のビットを有効にするもので、SL=1で有効になります。Rと合わせて使うと、L₀～L₂で指定した割り込みレベルの優先順位が最低に変更されます。また、EOIと合わせて使うと、特定のレベルにEOIを指定して割り込みをかけることができます。

D₅: EOI……割り込みの終了を宣言するコマンドで、EOI=1で終了を宣言します。SL=0の場合は、その時点で最も優先順位が高い割り込むレベル(現在処理中の割り込み処理)の終了を宣言します。

D₄: D₄=0, D₃: D₃=0……OCW₂を書き込む場合の設定です。

D₀～D₂: L₀～L₂……割り込みレベルを指定することができます。SL=1のときのみ有効です。

▶ OCW₃

アドレス: 00H(マスタ),

08H(スレーブ)/ライト

このワードは、読み出しレジスタ(IRR/ISR)の設定、完全ネスト・モード設定、ポーリングを行うときに用います。

D₆: ESMM, D₅: SMM……ESMM=1のとき、SMM=1なら完全ネスト・モード設定、SMM=0なら完全ネスト・モード解除します。ESMM=0のときは何もしません。

D₄: D₄=0, D₃: D₃=1……OCW₃を書き込み場合の設定です。

D₂: P……このビットは、ポーリング・モードへの移行のときに使用します。ポーリングは割り込み処理が未対応なCPU等を使用するときに使うもので、PC98シリーズではあまり有効ではありません。CPUの割り込みを禁止して、ポーリング・データ・ポート(図2-6)を見て割り込みがかっていることを検知します。

D₁: RR/D₀: RIS……RR=1のとき、RIS=1でISR、RIS=0でIRRを読み出すレジスタとして設定します。RR=0のときは何もしません。読み出せるI/Oアドレスは、マスタは00H、スレーブは08Hです。ポーリング・モードのときはポーリング・データが読み出せます。

◆ 割り込み処理の実際

割り込み処理の具体的なプログラムを紹介しておきます。この手のプログラムはアセンブラのほうが記述しやすいのですが、最近ではC言語を使用することが多く、CPUの処理速度も高速になってきてC言語でも十分な速度が得られるようになってきたので、C言語で記述してみました(リスト2-1～リスト2-3)。

◆ 割り込みコントローラの初期化

8259のイニシャライズ用の、イニシャライズ・ワー

〈図2-6〉 ポーリング・データ

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
INT	0	0	0	0	PL ₂	PL ₁	PL ₀

INT(Interrupt)=INT端子と同じ意味で、“1”のときμPD71059が、あるINTPを受け付けたことを示す
PL₂～PL₀(Permitted Level)=INTビットが“1”のときに有効で、受け付けた割り込みレベルを示す

ドには四つあります(図2-7)。これを指定されたI/Oアドレス・ポートに順番に書き込むことで、8259を初期化します。また、初期化はいつでも可能で、最初のイニシャライズ・ワード(ICW₁)を書き込むと、ICW₁に基づいたイニシャライズ・シーケンスが次のように開始されます。

- (1) 割り込み要求入力端子のエッジ・トリガ回路がリセットされ、トリガ・モードの場合はIRRがクリアされます。
- (2) ISR, IMRがクリアされます。
- (3) 割り込み優先順位が決定されます(最高:0→7:最低)。
- (4) 完全ネスト・モードは解除され、読み出しレジスタはIRRに設定されます。
- (5) ICW₄レジスタがクリアされ、通常ネスト・モード、非バッファ・モード、FIコマンド・モード、CALLモードが設定されます。

ICW₁を書き込んだ後は、次に必ずICW₂以降を書き込みます。

初期設定の注意としては、PC98シリーズのハードウェアに合ったように設定しないと正常な動作ができなくなります。

● ICW₁での注意

▶ D₀: IC₄(初期値=1)

ICW₄は、8080/8085モードと8086/8088モードを設定しなくてはならないので必ず必要です。

▶ D₁: SNGL(初期値=0)

8259はカスケード接続で二つついていますので「拡張モード」に設定します。

▶ D₂: ADI

8086/8088モードでは、この値は無効です。

▶ D₃: LTIM(初期値=0)

PC98シリーズでは、通常エッジ・トリガ・モードで使用します。エッジ・トリガ・モードは、割り込み要求信号の立ち上がりで割り込みがかかります。レベル・トリガ・モードでは、割り込み要求信号がハイ・レベルの間は何度でも割り込みがかかります。

▶ D₅～D₇: A₅～A₇

8086/8088モードでは、この値は無効です。

● ICW₂での注意

▶ D₀～D₂: A₈～A₁₀

```

/*
**
*/
#include <stdio.h>
#include <dos.h>

#define TIMER_VEC 0x08
#define TIMER_0 0x71
#define TIMER_MODE 0x77
#define PIC_IMR 0x02
#define PIC_OCW1 0x02
#define PIC_OCW2 0x00
#define EOI 0x20

int count = 0;
int endflag = 0;

void settimer(void);
void endtimer(void);
void interrupt timer(void);
void interrupt (*oldtimer)();

int main()
{
    unsigned int i, j, dmy=0;

    settimer();
    printf("簡易ベンチマーク (5秒お待ち下さい) >");
    for (i = 0; i < 60000; i++) {
        for (j = 0; j < 0x1fff; j++) {
            dmy += peek(j<<3, 0);
            if (endflag != 0) break;
        }
        if (endflag != 0) break;
    }
    printf("%u\n", i);
    endtimer();
    return 0;
}

/*
** インターバルタイマーの設定
**
*/
void settimer(void)
{
    int c10ms[2] = {24576, 19968};
    int sysclock;

    sysclock = (peekb(0, 0x0501)&0x80)>>7;

    outportb(PIC_OCW1, inportb(PIC_IMR)|0x01);
    oldtimer=getvect(TIMER_VEC);
    setvect(TIMER_VEC, timer);

    outportb(TIMER_MODE, 0x36);
    outportb(TIMER_0, c10ms[sysclock]&0xfff);
    outportb(TIMER_0, c10ms[sysclock]>>8);
    outportb(PIC_OCW1, inportb(PIC_IMR)&0xfe);
}

/*
** インターバルタイマーの終了
**
*/
void endtimer(void)
{
    outportb(PIC_OCW1, inportb(PIC_IMR)|0x01);
    setvect(TIMER_VEC, oldtimer);
}

/*
** インターバルタイマー割り込み部 (10ms)
**
*/
void interrupt timer(void)
{
    count++;
    if (count == 500) endflag = 1;
    outportb(PIC_OCW2, EOI);
}

/*
** システムクロック
** 割り込みマスク
** 以前のベクタ保存
** ベクタ設定
** #0 モード3
** カウンタ下位設定
** カウンタ上位設定
** 割り込み許可
*/

```


＜リスト 2-2＞

8259(スレーブ)を使用する場合の割り込み処理のプログラム

スレーブの割り込み処理

```
void interrupt slave_int(void)
{
    user_routine();                /* 割り込み処理ルーチン */

    outportb(0x08, 0x20);          /* スレーブにEOIを送る */
    outportb(0x08, 0x0b);          /* スレーブのISRを読む */
    if (inportb(0x08) == 0) {      /* 他の割り込みが無い? */
        outportb(0x00, 0x20);      /* マスターにEOIを送る */
    }
}
```

＜リスト 2-3＞

IRR/ISR 読み出し

IRRの読み方

マスター
outportb(0x00, 0x0a);
irr = inportb(0x00);

スレーブ
outportb(0x08, 0x0a);
irr = inportb(0x08);

ISRの読み方

マスター
outportb(0x00, 0x0b);
isr = inportb(0x00);

スレーブ
outportb(0x08, 0x0b);
isr = inportb(0x08);

IMRの操作

割り込み禁止

マスター
outportb(0x02, inportb(0x02)&maskbit);

スレーブ
outportb(0x0a, inportb(0x0a)&maskbit);

割り込み許可

マスター
outportb(0x02, inportb(0x02)&~maskbit);

スレーブ
outportb(0x0a, inportb(0x0a)&~maskbit);

8086/8088 モードでは、この値は無効です。

▶ D₃～D₇: T₃～T₇

割り込みベクタ番号の上位5ビットを指定します。下位3ビットは、レベルに合わせて自動的に0～7が割り当てられます。PC98シリーズでは下記のように設定してください。

00001×××=マスタ(08H～0FHを指定)

00010×××=スレーブ(10H～17Hを指定)

● ICW₃での注意

このイニシャライズ・ワードは、マスタとスレーブで設定データが異なります。マスタでは、スレーブがつながっているレベルを指定します。スレーブでは、つながっているマスタのレベル番号を指定します。PC98シリーズでは以下のように固定されています。

マスタ 80H INTP₇のみスレーブと接続
スレーブ 03H スレーブ番号は#7

● ICW₄での注意

▶ D₀: μPM(初期値=1)

8080/8085 モードと、8086/8088 モードを設定します。PC98シリーズでは**8086/8088 モードに設定**します。

▶ D₁: AEOI(初期値=0)

一般的には、割り込みの終了を8259に知らせるためにはEOIコマンドを書き込みます。しかし、このビットを1にすることで、自動EOIモードとすることができます。これはCPUが割り込み受け付けした時点で自動的にEOIコマンドを出力するもので、プログラムからEOIコマンドを出力する必要がありません。反面、割り込みプログラムが処理中にでも、同じ割り込みを許可してしまうので、割り込み処理が不安定になる可能性があります。PC98シリーズでは**AEOIは使用しません**。

▶ D₂: M/S

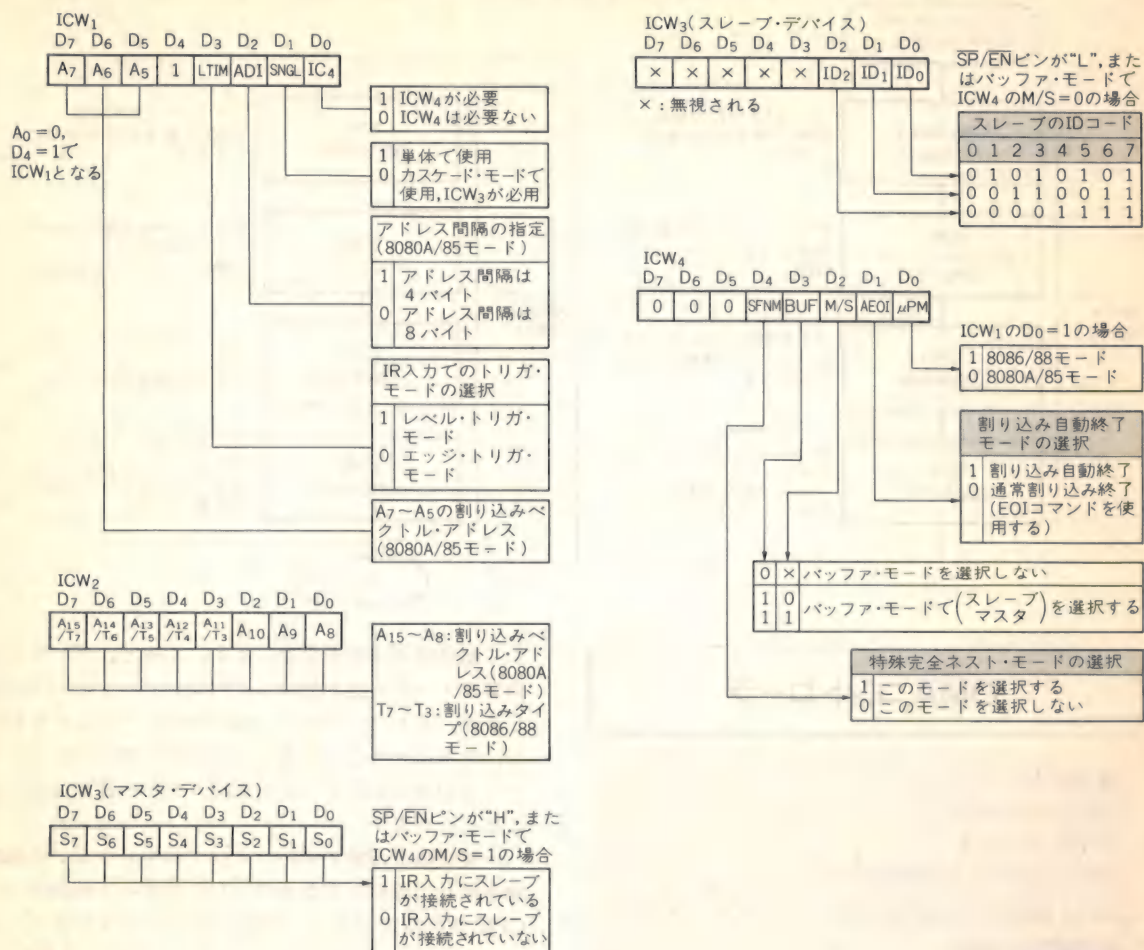
対象となる8259が、マスタであるかスレーブであるかを示します。**“1”でマスタ**、**“0”でスレーブ**となります。このビットが意味を持つのはバッファ・モードのときだけです。

▶ D₃: BUF(初期値=1)

8259とCPUとの接続状態がバッファ・モードであるかどうかを指定します。

CPUに接続される素子が多くなると、データ・バスにバッファをかける必要が生じます。8259は通常のデータ入出力動作以外に、割り込み要求が受け付けられたときに、データ・バス経由で割り込みを要求した

〈図 2-7〉 8259A の初期設定コマンド・ワード (ICW) のプログラム・フォーマット



デバイス, または処理の種類を示す割り込みベクタを CPU に送ります。このときデータ・バス・バッファの制御を 8259 の SP/EN ピンによって行います。この制御を行う場合, BUF=1 としてバッファ・モードを指定します。PC98 シリーズでは **バッファ・モードで** 使用します。

▶ D₄: SFNM (初期値: 1=マスタ/0=スレーブ)

SFNM=1 で特殊完全ネスト・モードになります。

PC98 シリーズでは, マスタは特殊完全ネスト・モードで, スレーブは完全ネスト・モードに指定します。

■ 初期化の手順

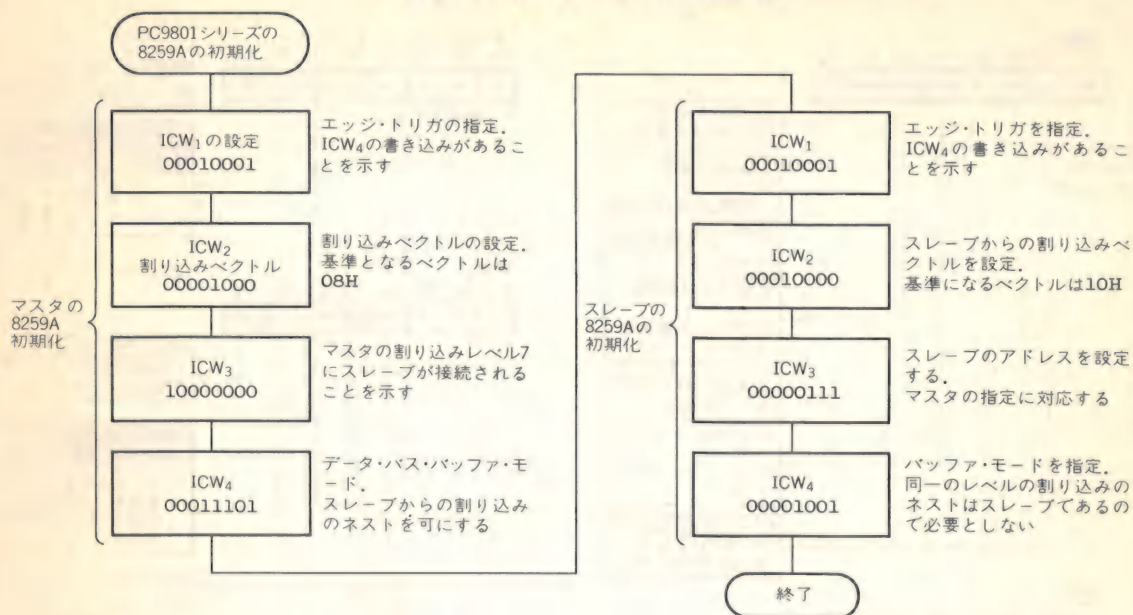
PC98 シリーズでの初期化の手順を図 2-8 に示します。8259 は周辺のハードウェアの状況に応じて, 初期化コマンドが限定されますので, それに合わない設定をした場合, 動作しなくなる場合がありますので注意が必要です。

初期設定の詳細は以下のようになっています。

- 8086/8088 モード
- カスケード接続の「拡張モード」
- エッジ・トリガ・モード
- 割り込みベクタ設定
 - マスタ 08H~0FH を指定
 - スレーブ 10H~17H を指定
- マスタとスレーブの接続状況を設定
 - マスタ INT_{P7}のみスレーブと接続
 - スレーブ スレーブ番号は#7
- 自動 EOI 禁止 (割り込み処理終了)
- バッファ・モード指定
- 特殊完全ネスト・モード (マスタのみ)

一般的には, システム本体で用意されたデバイスを使う場合には, 8259 はすでに初期化されていますから, 改めて初期化したり, 再設定したりする必要はほとんどありません。

〈図 2-8〉 PC9801 シリーズでの初期化の手順



DMA コントローラ

▶使用 LSI

μPD8237A 相当

▶I/O アドレス

01H~1FHの奇数番地(8237)

21H, 23H, 25H, 27H, 29H(バンク・レジスタ)

▶使用割り込み

なし

▶初期設定命令

コマンド・レジスタ=44H(DMA 許可)

■ DMA コントローラの役目と動作

DMA コントローラ(以下, DMAC と略す)は CPU を使用せずに, フロッピー・ディスク・インターフェース等の I/O デバイスから, メモリへ直接データを転送します。CPU を介在させて転送する場合は, デバイスがデータを持っているかどうかを確認するためにポーリングしたり, 割り込み等を使うために時間がかかります。さらに, メモリに転送する際には, 書き込み先アドレスの管理が必要です(図 2-9)。

DMAC の場合は, とても高速(約 800K バイト/秒)です。I/O デバイスがデータを持つと, そのデバイスが DMAC に対して転送要求信号を発生します。DMAC は, それを見て CPU にバス・ホールド信号を出して, CPU を停止させます。CPU の停止を確認したうえで, DMAC が転送要求を出したデバイスに対

して転送許可信号を発生します。すると, デバイスは自分のデータを誰も使用していないデータ・バスに流します。そして, DMAC が転送先のアドレスを出力し, さらにメモリへの書き込み信号を発生すれば, デバイスが発生したデータが直接メモリへ書き込まれることになります。

一見複雑な行程ですが, すべてハードウェアの専用線を使って行われるために, I/O アクセス時間とか, CPU 実行時間がなく, 問題になるのは RAM のアクセス時間だけとなります。98 の場合, 実際には DMAC の動作クロック(2.5~5 MHz)が遅く, 最近の高速な CPU に負けてしまったりします。

高速化されない DMAC に見切りを付けて, 最近ではインターフェース・カード自身に専用の DMAC を持たせたもの(バス・マスタ)も登場しています。

■ DMA コントローラのハードウェア

PC98 シリーズの DMAC には μPD8237A 相当品(以下 8237 と略す)を使用しています。これは 16 ビットのアドレス・バスと 8 ビットのデータ・バスを持ち, 単体では 64K バイトまでのメモリ・アクセスを行えます。上位 8 ビット分のアドレス・バス(A₈~A₁₅)と, データ・バスは兼用になっており, 上位 8 ビット分のアドレスを TTL 等でラッチして使用しています。また, 優先順位が付いた 4 チャンネル分の DMA を持っています。

DMAC の動作中は, アドレス・バスが出力となりデータの転送を行います。DMAC にコマンドやパラメータを与えるときには, 下位アドレス・バスの 4 本

〈図2-9〉 DMA コントローラの I/O アドレス一覧

	命 令	READ/ WRITE	I/O ポート ・ アドレス	デ ー タ							
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
DMAC 8237	チャンネル 0 アドレス	R/W	01	A ₇ A ₁₅	A ₆ A ₁₄	A ₅ A ₁₃	A ₄ A ₁₂	A ₃ A ₁₁	A ₂ A ₁₀	A ₁ A ₉	A ₀ A ₈
	チャンネル 0 カウント	R/W	03	C ₇ C ₁₅	C ₆ C ₁₄	C ₅ C ₁₃	C ₄ C ₁₂	C ₃ C ₁₁	C ₂ C ₁₀	C ₁ C ₉	C ₀ C ₈
	チャンネル 1 アドレス	R/W	05	A ₇ A ₁₅	A ₆ A ₁₄	A ₅ A ₁₃	A ₄ A ₁₂	A ₃ A ₁₁	A ₂ A ₁₀	A ₁ A ₉	A ₀ A ₈
	チャンネル 1 カウント	R/W	07	C ₇ C ₁₅	C ₆ C ₁₄	C ₅ C ₁₃	C ₄ C ₁₂	C ₃ C ₁₁	C ₂ C ₁₀	C ₁ C ₉	C ₀ C ₈
	チャンネル 2 アドレス	R/W	09	A ₇ A ₁₅	A ₆ A ₁₄	A ₅ A ₁₃	A ₄ A ₁₂	A ₃ A ₁₁	A ₂ A ₁₀	A ₁ A ₉	A ₀ A ₈
	チャンネル 2 カウント	R/W	0B	C ₇ C ₁₅	C ₆ C ₁₄	C ₅ C ₁₃	C ₄ C ₁₂	C ₃ C ₁₁	C ₂ C ₁₀	C ₁ C ₉	C ₀ C ₈
	チャンネル 3 アドレス	R/W	0D	A ₇ A ₁₅	A ₆ A ₁₄	A ₅ A ₁₃	A ₄ A ₁₂	A ₃ A ₁₁	A ₂ A ₁₀	A ₁ A ₉	A ₀ A ₈
	チャンネル 3 カウント	R/W	0F	C ₇ C ₁₅	C ₆ C ₁₄	C ₅ C ₁₃	C ₄ C ₁₂	C ₃ C ₁₁	C ₂ C ₁₀	C ₁ C ₉	C ₀ C ₈
	ステータス・リード	R	11	RQ ₃	RQ ₂	RQ ₁	RQ ₀	TC ₃	TC ₂	TC ₁	TC ₀
	デバイス・コントロール	W	11	AKL	RQL	EXW	ROT	CMP	DDMA	AHLD	MTM
	リクエスト・コントロール	W	13	—	—	—	—	—	SRQ	SELCH	
	マスク・コントロール	W	15	—	—	—	—	—	MSET	SELCH	
	モード・コントロール	W	17	TMODE		ADIR	SEFI	TDIR		SELCH	
	アドレス・ロー・バイト	W	19	—	—	—	—	—	—	—	—
	テンポラリ・レジスタ	R	1B	—	—	—	—	—	—	—	—
	ソフトウェア・リセット	W	1B	—	—	—	—	—	—	—	—
バンク・ レジスタ	クリア・オール・マスク ・レジスタ	W	1D	—	—	—	—	—	—	—	—
	マスク・コントロール・ レジスタ	W	1F	—	—	—	—	M ₃	M ₂	M ₁	M ₀
	チャンネル 1 バンク	W	21	A ₂₃	A ₂₂	A ₂₁	A ₂₀	A ₁₉	A ₁₈	A ₁₇	A ₁₆
	チャンネル 2 バンク	W	23	A ₂₃	A ₂₂	A ₂₁	A ₂₀	A ₁₉	A ₁₈	A ₁₇	A ₁₆
	チャンネル 3 バンク	W	25	A ₂₃	A ₂₂	A ₂₁	A ₂₀	A ₁₉	A ₁₈	A ₁₇	A ₁₆
	チャンネル 0 バンク	W	27	A ₂₃	A ₂₂	A ₂₁	A ₂₀	A ₁₉	A ₁₈	A ₁₇	A ₁₆
	バンク・アドレス・オート・ インクリメント・モード・ レジスタ	W	29	0	0	0	0	M ₁	M ₀	CS ₁	CS ₀

がアドレス入力用になり、レジスタの選択をします。DMAC は 16 個の I/O アドレスに 26 個のレジスタを持っています。

98 シリーズでは、DMAC の I/O アドレスを奇数アドレスの、01H, 03H, 05H, … 1FH の 16 個に割り当てていますので、DMAC のレジスタを CPU がアクセスする時は、DMAC のアドレス・バスと CPU のアドレス・バスが、A₀=A₁, A₁=A₂のように、一つ分ずれて接続されています。もちろん、メモリ転送する場合は、奇数アドレスも偶数アドレスも両方アクセスできなくてはなりませんから、A₀=A₀, A₁=A₁のような接続に切り替えます。

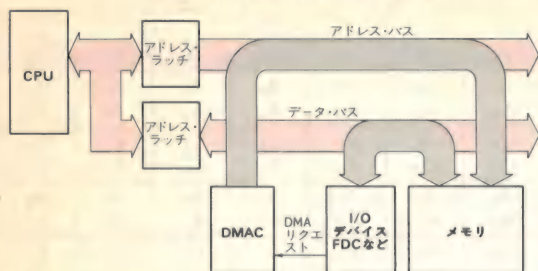
DMAC を使用する I/O デバイスは、フロッピー・ディスクやハード・ディスクのコントローラですが、それらは I/O 空間の偶数アドレス(データ・バスの下位 8

ビット)に接続されています。DMAC を使用する場合は、図 2-10 のように、I/O デバイスが DMAC に DMA のリクエスト信号を送ります。すると、DMAC はアドレスを発生させます。そのときに I/O デバイスとメモリの間でデータ・バスを通して直接にデータ転送が行われます。

ここで、フロッピー・ディスク・コントローラ等は、データ・バスの下位 8 ビットに接続されていますので、そのままでは奇数アドレス(上位)のメモリとは、データ・バスが直接接続されていませんので転送ができません。

そのため、図 2-11 のような回路がデータ・バスに接続されていて、下位 D₀~D₇と、上位 D₈~D₁₅とを結んで、奇数アドレスと偶数アドレスとのデータ転送を可能にしています。奇数アドレスと偶数アドレスの転

〈図 2-10〉 DMA 転送のしくみ



送の違いを図 2-12 に示します。

図 2-11 の回路が、奇数アドレスを持つメモリ・アクセス時に介在してくるので、DMA を使用するデバイスはすべて偶数アドレスにする必要があります。また、メモリとメモリの間の転送は、DMAC のレジスタが奇数アドレスにあるため使用できません。

■ DMA コントローラのレジスタ

DMAC のレジスタは大きく分けると、アドレス・レジスタ、カウント・レジスタ、コントロール・レジスタ群に分けられます(図 2-13)。

● アドレス/カウント・レジスタ

▶ 01H, 05H, 09H, 0DH: アドレス・レジスタ/ライト

アドレス・レジスタは、DMA 転送用アドレスの初期値を入れます。このレジスタは一つの I/O アドレスに 2 回書き込むことで、16 ビット分を指定することができます。

▶ 01H, 05H, 09H, 0DH: イフェクティブ・アドレス・レジスタ/リード・ライト

DMA 転送中に有効なレジスタです。

DMA 転送中のアドレスを読み書きできます。DMA が転送するたびに自動的に+1/-1 されていきます。このレジスタは一つの I/O アドレスに 2 回読み書きすることで、16 ビット分を指定することができます。

▶ 03H, 07H, 0BH, 0FH: カウント・レジスタ/ライト

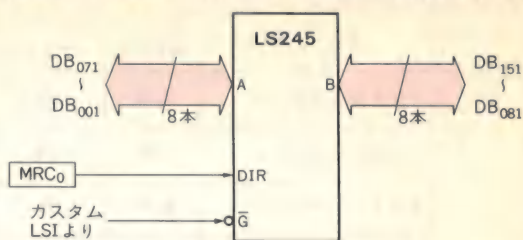
DMA 転送バイト数の初期値を入れます。転送開始時に、このレジスタの内容をイフェクティブ・カウント・レジスタへロードします。再度、CPU から書き換えるまでは値は変わりません。このレジスタは一つの I/O アドレスに 2 回読み書きすることで、16 ビット分を指定することができます。

▶ 03H, 07H, 0BH, 0FH: イフェクティブ・カウント・レジスタ/リード・ライト

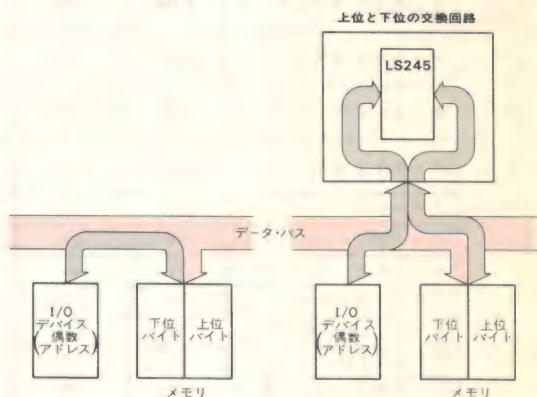
DMA 転送中に有効なレジスタです。

DMA 転送の残りバイトを読み書きできます。DMA が転送するたびに自動的に-1 されていき、0

〈図 2-11〉 データ・バスの上位と下位を結ぶ双方向性バッファ



〈図 2-12〉 奇数アドレスと偶数アドレスの転送の違い



からアンダ・フローした場合に、DMA 転送終了となります。このレジスタは一つの I/O アドレスに 2 回読み書きすることで、16 ビット分を指定することができます。

▶ 11H: デバイス・コントロール・レジスタ/ライト

DMA 転送における転送モード、DMA 要求の許可/禁止等の制御に使用します。図 2-14 にフォーマットを示します。

D₀: MTM(初期値=0)……MTM=1 でメモリ-メモリ間転送を行います。メモリ間転送はチャンネル 0 → 1 に固定され、ソフトウェア DMA 要求でスタートします。ただし、PC98 シリーズではメモリ間転送はできませんので、通常は MTM=0 と設定します。

D₁: AHLD(初期値=0)……MTM=1 のときに、AHLD=1 にすると、チャンネル 0 のアドレスは初期値に固定されます(自動インクリメントされない)。MTM=0 のときは無視されます。

D₂: DDMA……DDMA=1 で、すべての DMA 要求を禁止します。

D₃: CMP(初期値=0)……CMP=1 で圧縮転送を行います。圧縮転送は、DMAC の上位アドレスの出力を必要と時のみ行うことで、転送のサイクル(タイミング)を縮めて高速に転送する方式です。ただし、ディマインド・モード/ブロック・モードのみ有効なので、PC98 シリーズでは使用せず、通常転送(CMP=0)で使

D₄: ROT(初期値=0)……ROT=1 で回転ネスト・モード, ROT=0 で固定ネスト・モードです。回転ネスト・モードは、最後に DMA サービスを受けたチャンネルの優先順位が最低になるように順次変化させて、特定のチャンネルが独占されるのを防ぎます。PC98 は、優先順位が「0>1>2>3」で固定されている固定ネスト・モードで使します。

D₅: EXW(初期値=0)……EXW=1 で拡張書き込みモード, EXW=0 で遅れ書き込みモードです。EXW=1 にすると、メモリや I/O 書き込みタイミングが、読み込みタイミングと同じタイミングで出力されます。場合によっては、ウェイト・サイクルを減らすことができますが、PC98 シリーズでは通常使用しません。また、CMP=1 の場合は無効になります。

D₆: RQL(初期値=1)……DRQ(DMA 要求信号)アクティブ・レベルを変更します。RQL=0 でアクティブ“H”, RQL=1 でアクティブ“L”です。PC98 シリーズでは、RQL=1 で使します。

D₇: AKL(初期値=0)……DACK(DMA 許可信号)アクティブ・レベルを変更します。AKL=0 でアクティブ“H”, AKL=1 でアクティブ“L”です。PC98 シリーズでは、AKL=0 で使します。

▶ 11H: ステータス・リード・レジスタ/リード

各チャンネルの、DMA 要求、転送終了状態を調べます。図 2-15 にステータス・リード・レジスタのフォーマットを示します。

D₀~D₃: TC₀~TC₃……END 入力、またはターミナル・カウント(転送終端)の場合は TC_x=1, 未終端の場合は TC_x=0 になります。

D₄~D₇: RQ₀~RQ₃……DRQ 端子によって起きた、DMA 要求の状態を示します。DMA 要求があった場

〈図 2-13〉 アドレス/カウント・レジスタ

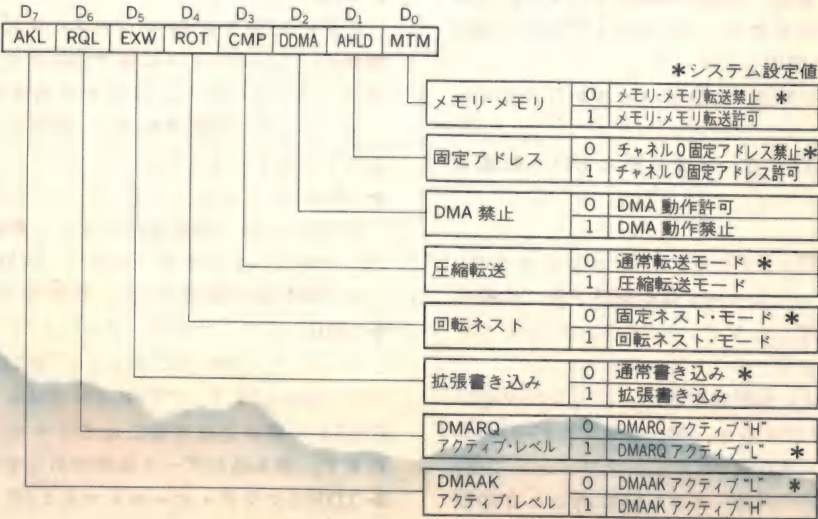
チャンネル	レジスタ名称	READ/WRITE	I/O アドレス
0	セット・アドレス・レジスタ	W	01H
	イフェクティブ・アドレス・レジスタ	R/W	
	セット・カウント・レジスタ	W	03H
	イフェクティブ・カウント・レジスタ	R/W	
1	セット・アドレス・レジスタ	W	05H
	イフェクティブ・アドレス・レジスタ	R/W	
	セット・カウント・レジスタ	W	07H
	イフェクティブ・カウント・レジスタ	R/W	
2	セット・アドレス・レジスタ	W	09H
	イフェクティブ・アドレス・レジスタ	R/W	
	セット・カウント・レジスタ	W	0BH
	イフェクティブ・カウント・レジスタ	R/W	
3	セット・アドレス・レジスタ	W	0DH
	イフェクティブ・アドレス・レジスタ	R/W	
	セット・カウント・レジスタ	W	0FH
	イフェクティブ・カウント・レジスタ	R/W	

合に RQ_x=1 です。

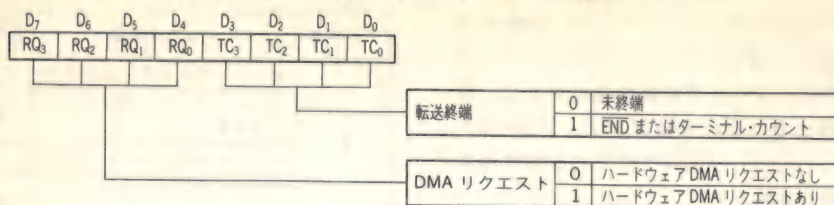
▶ 13H: リクエスト・コントロール・レジスタ/ライト (使用禁止)

ソフトウェアによる DMA 要求を実行します。しかし、PC98 シリーズでは、ソフトウェアによる DMA 要求は禁止されています。図 2-16 にリクエスト・コントロール・レジスタのフォーマットを示します。

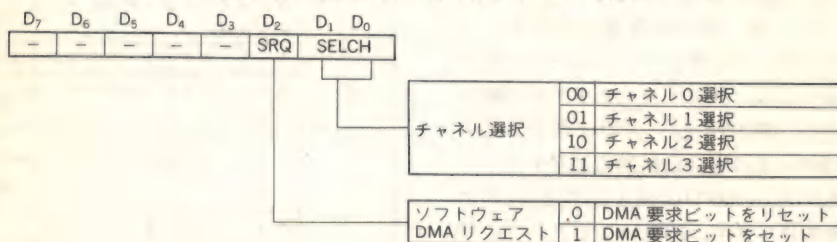
〈図 2-14〉 デバイス・コントロール・レジスタ



〈図 2-15〉 ステータス・リード・レジスタ



〈図 2-16〉 リクエスト・コントロール・レジスタ



▶ 15H: マスク・コントロール・レジスタ/ライト

図 2-17 に示すこのレジスタは、ハードウェア (DRQ 端子) からの **DMA 要求の許可/禁止を設定**します。設定には、チャンネルごとに設定する方法と、すべてのチャンネルを同時に設定する方法がありますが、「15H: マスク・コントロール・レジスタ」はチャンネルごとに別々に設定します。

DMAC にコマンドを与える場合は、DMAC が動作中であってはいけません。そのために、コマンドを与える前にマスク・ビットをセットしておきます。

▶ 17H: モード・コントロール・レジスタ/ライト

各チャンネルごとに、DMA 転送動作を制御します (図 2-18)。

D₀~D₁: SELCH……DMA チャンネルを指定します (0~3)

D₂~D₃: TDIR……転送方向を指定します。

00=ペリファイ転送。DMA は動作しますが、**実際にメモリへ転送されません**。フロッピー・ディスク等の CRC チェック等に使用します。

01=I/O → メモリ転送。I/O デバイスからメモリへ転送します。

10=メモリ → I/O 転送。メモリから I/O へ転送します。

11=使用禁止

D₄: SEFI……SEFI=1 でセルフ・イニシャライズ・モードになります。このモードは**転送終了後に自動的に、最初に初期設定したアドレス/カウンタ・レジスタの内容をロード**します。

D₅: ADIR……DMA が転送する際に、カウンタをインクリメント (+1) させるか、デクリメント (-1) させるかを指定します。ADIR=0 でインクリメントです。D₆~D₇: TMODE……I/O-メモリ間転送のときの転

送モードを選びます。PC98 シリーズでは「**01: シングル・モード**」を使用します。

00=デマインド・モード。転送がすべて終了するか、DMA 要求信号がクリアされるまで転送を続けます。その間、CPU は停止しているのでプログラムは一切動きません (割り込みも同様)。

01=シングル・モード。1 回 (1 バイト) 転送するたびに、CPU が 1 マシン・サイクル動作します。**DMA 転送中であっても割り込み等が受け付けられます**。

10=ブロック・モード。転送がすべて終了するまで転送を続けます。その間、CPU は停止しているのでプログラムは一切動きません。メモリ間転送はブロック・モードで行います。

11=拡張モード。DMAC をカスケード接続して使う場合に使用しますが、PC98 シリーズでは、DMAC は一つありませんから使用しません。

▶ 19H: アドレス・ロウ・バイト/ライト

アドレス/カウンタ・レジスタに**新たな値を設定する場合に、このアドレスに書き込み**ます。このアドレスはレジスタではなく、CPU から書き込みすることでコマンドとして認識されます。書き込むデータは何であってかまいません。

▶ 1BH: テンポラリ・データ・レジスタ/リード

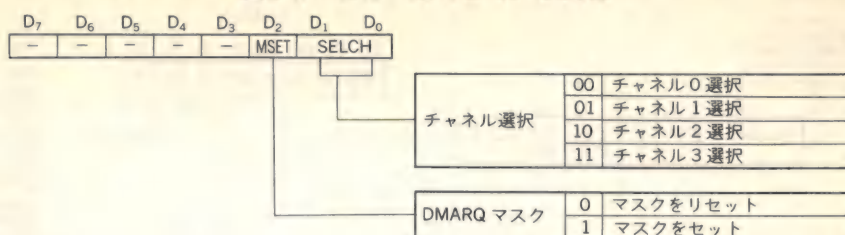
メモリ-メモリ間転送のときに、最後に転送されたデータが入っています。ただし、PC98 シリーズでは**メモリ間転送は無効なので、無意味**です。

▶ 1BH: ソフトウェア・リセット/ライト

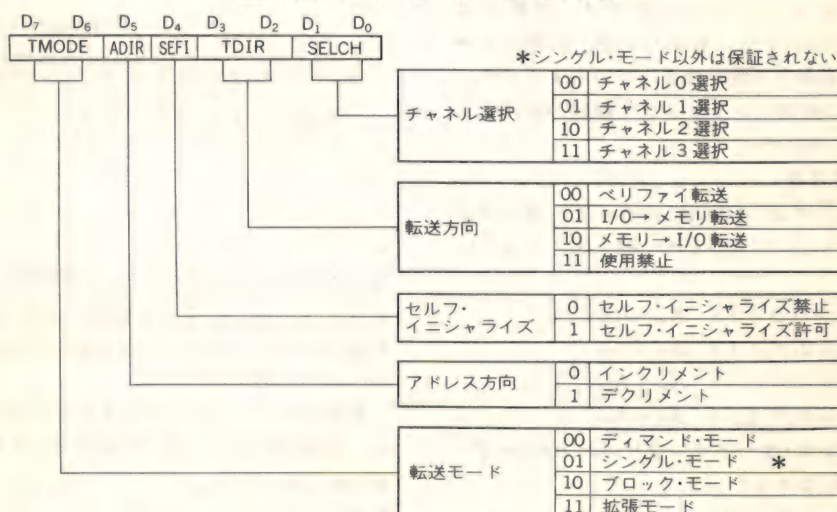
ハードウェアからのリセット同様に、**DMAC にリセット**をかけます。このアドレスはレジスタではなく、CPU から書き込みすることでコマンドとして認識されます。書き込むデータは何であってかまいません。

▶ 1DH: クリア・オール・マスク/ライト

〈図 2-17〉 マスク・コントロール・レジスタ



〈図 2-18〉 モード・コントロール・レジスタ



*シングル・モード以外は保証されない

全チャンネルのマスク・コントロール・レジスタをクリアし、DMA 転送の要求を許可します。このアドレスはレジスタではなく、CPU から書き込みすることでコマンドとして認識されます。書き込むデータは何であってかまいません。

▶ 1FH: マスク・コントロール・レジスタ/ライト
ハードウェア (DRQ 端子) からの DMA 要求の許可/禁止を設定します。このレジスタからは、すべてのチャンネルを同時に設定します (図 2-19)。

◆ 64K バイトを超えるアクセス

DMAC 単体では 64K バイトまでのアクセスしかできません。それ以上のアクセスを可能にするために、A₁₆~A₁₉ までの 4 ビットを拡張したバンク・レジスタがあります。このレジスタは、CPU からレジスタに書き込んだデータを、DMAC がメモリ・アクセスする際に、そのままアドレス・バスに出力し、1M バイトまでのアクセスを可能にしたものです。

CPU に V30 を持つ機種では、DMAC とバンク・レジスタとは無関係に動作しますので、64K バイトの境界を越えるアクセスはできません。この場合は 2 回に分けて、バンク・レジスタを設定しなおして転送する

ことになります。

CPU に 80286 以上を持つ機種では、バンク・レジスタが A₁₆~A₂₃ までの 8 ビットに拡張され、16M バイトまでのメモリがアクセス可能です。さらに、64K バイトの境界で、自動的にバンク・レジスタがインクリメントするので、すべて、一度で転送できるようになりました。

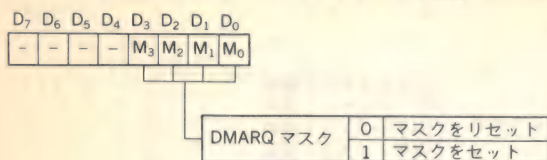
バンク・レジスタが 4 ビットから 8 ビットへ拡張されたのに従って、互換を持たせるために通常は 8 ビットのレジスタの上位 4 ビットが、“0” にマスクされています。1M バイトを超える転送をするときには、DMA アドレス・マスク・レジスタ (0439H) を操作することで可能になります。

● DMA アドレス・マスク・レジスタ (0439H/リード・ライト)

図 2-20 にこれを示します。D₂ 以外のビットも他の機能に使われていますので、MSK ビットを変更する場合は、D₀~D₇ まですべてのビットを読み取り、MSK ビットのみ変更して、再び書くようにします。また、DMA 転送が終了次第、MSK ビットを元の状態に戻す必要があります。

PC9821Af 以降の機種では、メモリ空間を 16M バ

〈図 2-19〉 マスク・コントロール・レジスタ



イト以上持てるものがあります。これらの機種では、バンク・レジスタが8+8ビットに拡張されていて、全部で32ビットまでアクセス可能です。計算上は4096Mバイトまでアクセス可能ですが、物理的なメモリ増設の最大容量で上限が決まっているようです。しかし、DMAの自動インクリメント機能が動作するのは16Mバイトだけです。

● バンク・レジスタ

バンク・レジスタは、DMACのチャンネル分だけあり、I/Oアドレスは、以下のように割り当てられています。

21H チャンネル1(ライト) A₁₆~A₂₃

23H チャンネル2(ライト) A₁₆~A₂₃

25H チャンネル3(ライト) A₁₆~A₂₃

27H チャンネル0(ライト) A₁₆~A₂₃

● バンク・アドレス・オートインクリメント・モード・レジスタ (29H/ライト)

図 2-21 に示します。

◆ DMA コントローラの使用状況

DMAの使用状況は、ノーマル・モード時ではどの機種でも同じような割り当てがされています。異なるのは、ハード・ディスク・ドライブ・インターフェースや、メモリ・リフレッシュに使用している機種で少し異なります。DMAの優先順位は「0」が一番高く、「3」が一番低くなり、同時に発生した場合は優先順位が高い順番に処理されます。

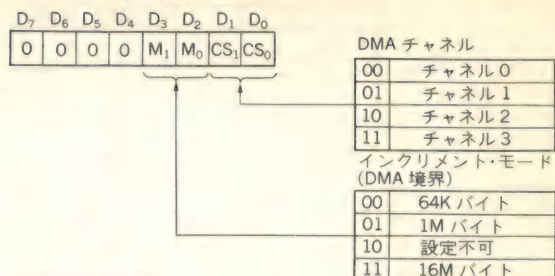
▶ チャンネル#0

主にハード・ディスク・ドライブ・インターフェースに使われます。ディップ・スイッチ SW₃₋₃で、チャンネルを「0」、「1」に替えられるものもあります。

▶ チャンネル#1

主に未使用か使用不可のどちらかで、内蔵ハード・ディスク・ドライブ・インターフェースに使われている

〈図 2-21〉 バンク・アドレス・オートインクリメント・モード・レジスタ



PC-98XA では 11 の設定は不可

インクリメント・モード: 8237 から出力されるアドレスと連結してバンク・アドレスをインクリメントする

00: バンク・アドレスをインクリメントしない

01: 1M バイトまでのインクリメント

11: 16M バイトまでのインクリメント

ものもあります。また、古い機種では、メモリ・リフレッシュに使われている機種もあります(PC9801/E/F/M/U/VF/VM/UV, PC286U/L/LE/NOTE executive/NOTE F)。

拡張スロットにチャンネル#1の信号が出ていないため、拡張基板からではDMAを使用できません(PC98XAは例外)。

▶ チャンネル#2

2HD フロッピー・ディスク・インターフェースに使われています。

拡張スロットにチャンネル#2の信号が出ていないため、拡張基板からではDMAを使用できません(PC9801/E/F/U/VF/VM/UV/CVでは最も大きな番号のスロットのみ使用可能。PC98XAは例外)。

▶ チャンネル#3

2DD フロッピー・ディスク・インターフェースに使われています。

チャンネル割り当てを図 2-22 に、使用状況を図 2-23 に示します。

◆ DMA コントローラの使用方法

DMA 使用方法のサンプル・プログラムとして、1MB フロッピー・ディスク・ドライブを読み出すプログラムを紹介します(リスト 2-4)。

〈図 2-20〉

DMA アドレス・マスク・レジスタ

命 令	READ/ WRITE	I/O ポート ・ アドレス	デ ー タ							
			D7	D6	D5	D4	D3	D2	D1	D0
DMA アドレス・ マスク・レジスタ	R/W	0439	-	-	-	-	-	MSK	-	-

D₂: MSK0: A₂₃~A₂₀有効

1: A₂₃~A₂₀無効

〈図 2-22〉 チャネル割り当て

チャネル			0	1	2	3
PC9801/E/F1,2,3/M2,3/U2/ VF2/VM0,2,4/UV2/VM21			HDD	メモリ・ リフレッシュ*1	1MB FD	640KB FD
PC9801T/DA/DS/DX/CS/ FA/FS/FX/ PC9821Ap/As/Ae/Af/ PC98GS	SW ₃₋₃ OFF	HDD	使用不可	1MB FD	640KB FD	
	SW ₃₋₃ ON	未使用	内蔵 HDD	1MB FD	640KB FD	
PC98XA model 1,2,3/11,21,31			未使用	1MB FD	640KB FD	HDD
PC98XL model 1,2,4 PC98XL ²		ノーマル	HDD	未使用	1MB FD	640KB FD
		ハイレゾ	HDD	1MB FD	未使用	640KB FD
PC98RL*2	SW ₃₋₃ OFF	ノーマル	HDD	使用不可	1MB FD	640KB FD
		ハイレゾ	HDD	1MB FD	使用不可	640KB FD
	SW ₃₋₃ ON	ノーマル	未使用	内蔵 HDD	1MB FD	640KB FD
		ハイレゾ	未使用	1MB FD	内蔵 HDD	640KB FD
上記以外			HDD	未使用	1MB FD	640KB FD
優先順位			高 ←	→ 低		

* 1：メモリ・リフレッシュは 64K バイト単位で行う

* 2：ディップ・スイッチによる DMA のチャネル切り替えは、本体内蔵固定ディスクのみ可能

〈図 2-23〉 DMA チャネル使用状況

DMA チャネル		#0	#1	#2	#3
オプション・ボード					
PC9801-08/09/本体内蔵	640KB FD I/F				◎
PC9801-15/本体内蔵	1MB FD I/F			◎	
システム・リザーブ			◎		
PC9801-07/27/本体内蔵	HDD I/F	◎			
PC9801-29N	GP-IB I/F	○			◎
PC9801-36	CGMT I/F	○			◎
PC9801-37	ファクシミリ・ボード	○			◎
PC98XL-02	ImPP ボード	○			◎
PC9801-55/L/U/92	SCSI I/F	◎		○	○
PC9866/L	通信制御アダプタ	◎			○
PC9801-82	GP-IB ボード	○			○

◎：工場出荷時設定 ○：変更可能レベル

〈参考リスト〉 8253の動作中のカウン
ト・データの読み取り方法
(カウンタ・ラッチ・コマンドを使用す
ると、その瞬間のカウンタの内容が読み
取れる)

```

/*
** 8253 のカウンタデータ読取り
*/

#define counter0 0x71 /* 8253 couner #0 : timer */
#define counter1 0x73 /* 8253 couner #1 : beep */
#define counter2 0x75 /* 8253 couner #2 : rs232c */
#define mode8253 0x77 /* 8253 mode reg. */

#define latch0 0x00 /* counter #0 : count latch : binary */
#define latch1 0x40 /* counter #1 : count latch : binary */
#define latch2 0x80 /* counter #2 : count latch : binary */

unsigned int getcounter(void)
{
    unsigned int count;
    outportb(counter2, latch2); /* counter #2 */
    count=inportb(counter2);
    count=count | (inportb(counter2) << 8);
    return(count);
}

```



```

/* DMA テスト
**
** ドライブ 0, トラック 0, サイド 0, セクター 1~8 を DUMP 表示.
*/
#include <stdlib.h>
#include <stdio.h>
#include <dos.h>

#define FDC_STAT 0x90
#define FDC_RESULT 0x92
#define FDC_CMD 0x92

#define DMA_TFR_SIZE 1024*8
#define DMA_MASK 0x15
#define DMA_MODE 0x17
#define DMA_STAT 0x11
#define DMA_CLRP 0x19
#define DMA_CH2_BANK 0x23
#define DMA_CH2_ADRS 0x09
#define DMA_CH2_COUNT 0x0b

char *buff;

void setfdccmd(int *cmd);
void setdma(unsigned int tfrseg, unsigned int tfrsize);

void main(void)
{
    int fdc_seek[] = {0x0f, 0, 0, -1};
    int fdc_read[] = {0x46, 0, 0, 1, 3, 8, 0x35, 0xff, -1};
    int i, count;
    /* seek */
    /* read */

    if ((buff = calloc(DMA_TFR_SIZE, 1)) != NULL) {
        setfdccmd(&fdc_seek);
        setdma(FP_SEG(buff), FP_OFF(buff), DMA_TFR_SIZE);
        setfdccmd(&fdc_read);
        for (i = 0; i < DMA_TFR_SIZE; i++) {
            do {
                count = inportb(DMA_CH2_COUNT);
                count += inportb(DMA_CH2_COUNT) < 8;
            } while (count >= DMA_TFR_SIZE - 1);
            if (i%16 == 0) printf("%04X : ", i);
            printf(" %02X", buff[i]);
        }
        printf("\n");
        inportb(DMA_STAT);
        free(buff);
    }
    /* TC bit clear */
}

/* FDC にコマンドをセット
**
** コマンド (パラメータ) 終了は -1
**
void setfdccmd(int *cmd)
{
    int stat, i=0;
    while (cmd[i] >= 0) {
        do {
            stat = inportb(FDC_STAT);
        } while (((stat & 0x80) == 0) || ((stat & 0x40) != 0));
        outportb(FDC_CMD, cmd[i]);
        i++;
    }
}

/* DMA にパラメータ設定
**
**
void setdma(unsigned int tfrseg, unsigned int tfrsize, unsigned int tfrsize)
{
    unsigned int bank, adres;

    outportb(DMA_MASK, 0x06);
    bank = tfrseg >> 12;
    adres = (tfrseg << 4) + tfrsize;
    outportb(DMA_CH2_BANK, bank);
    outportb(DMA_CH2_ADRS, adres & 0xff);
    outportb(DMA_CH2_ADRS, adres >> 8);
    outportb(DMA_CH2_COUNT, tfrsize & 0xff);
    outportb(DMA_CH2_COUNT, tfrsize >> 8);
    /* カウント HI 設定
    モード設定 シフト転送, アト・レスインクリメント, オート・リロード,
    outportb(DMA_MODE, 0x46);
    outportb(DMA_MASK, 0x02);
}

```

タイマ

▶使用 LSI

8253C 相当

▶ I/O アドレス

71H, 73H, 75H, 77H (旧機種)

71H, 75H, 77H, 3FDBH, 3FDFH

▶使用割り込み

IR₀ (割り込みベクタ #08H)

図 2-24 に I/O アドレス一覧を示します。

98 シリーズでは、タイマ用 LSI に μ PD8253C の互換品 (以降は 8253 と略す) を使用しています。この LSI は、入力されたクロック (約 2~2.5 MHz/CLK 入力) を 16 ビットのカウンタで分周して任意の周波数を作り出すことができます。また、三つの独立した出力と三つの独立したカウンタを持ち、それぞれ六つのモードを選択することができます。

タイマ LSI 周辺回路を図 2-25 に示します。

◆ 8253 に入力されるクロック (CLK 入力) の違い

LSI に入力されるクロック (以下、CLK 入力と略

す) には 2 種類あり、拡張スロットのシステム・クロックが 8 MHz 系 (CPU クロックが、8/16/33/60/66 MHz) の時には、1.9968 MHz、5 MHz 系 (CPU クロックが、5/10/12/20/25/40 MHz) のときには 2.4576 MHz になります (機種によっては該当しない場合もある)。タイマ LSI はこのクロックを分周して使用していますから、8253 にどちらのクロックが入力されているか認識しないと、タイミングや周波数が変わってしまい互換性がとれなくなってしまいます。これを調べるには、プリンタ・ポート用の 8255 の 42H のビット 5 を読み取り、“1” ならば 1.9968 MHz、“0” ならば 2.4576 MHz であることがわかります。

EPSON の PC シリーズで、PC386S/G/GS/GE/P/GR/GF では、システム・クロックを変更できますので、それに合わせた CLK 入力が入ります。その他の機種では、CPU クロックが 10 MHz 以上ならば 2.4576 MHz (PC286VE の 10 MHz 時を除く)、10 MHz 未満ならば 1.9968 MHz が供給されます。NEC の 98 シリーズと若干異なりますが、プリンタ・ポート用の 8255 の 42H のビット 5 で得られる情報は同様ですから、プログラムの互換性が保たれます。

例外として、初代の PC9801 では、このプリンタ・ポート用の 8255 の 42H のビット 5 がプリンタ・インターフェース・コネクタの 13 ピンにつながっており、

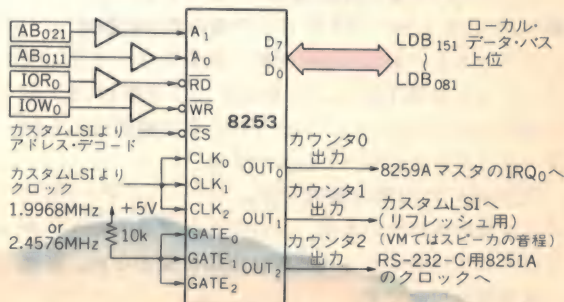
命 令	READ/ WRITE	I/O ポート ・ アドレス	デ ー タ							
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
カウンタ 0 へのロード	W	71	C ₇	—	—	—	—	—	—	C ₀
			C ₁₅	—	—	—	—	—	—	C ₈
カウンタ 0 をリード	R	71	C ₇	—	—	—	—	—	—	C ₀
			C ₁₅	—	—	—	—	—	—	C ₈
カウンタ 1 へのロード	W	73/3FDB * 1	C ₇	—	—	—	—	—	—	C ₀
			C ₁₅	—	—	—	—	—	—	C ₈
カウンタ 1 をリード	R	73/3FDB * 1	C ₇	—	—	—	—	—	—	C ₀
			C ₁₅	—	—	—	—	—	—	C ₈
カウンタ 2 へのロード	W	75	C ₇	—	—	—	—	—	—	C ₀
			C ₁₅	—	—	—	—	—	—	C ₈
カウンタ 2 をリード 0	W	75	C ₇	—	—	—	—	—	—	C ₀
			C ₁₅	—	—	—	—	—	—	C ₈
モード指定	W	77/3FDF * 1	SC ₁	SC ₀	RL ₁	RL ₀	M ₂	M ₁	M ₀	BCD

* 1: カウンタ 1 およびモード指定のアドレスは、次のようになる

	カウンタ 1	モード指定
PC9801/E/F1,2,3/M2,3		
PC98XA		
PC98XL ハイレゾ・モード動作時	73H	77H
PC98XL ² ハイレゾ・モード動作時		
PC98RL ハイレゾ・モード動作時		
上記以外	3FDBH	77H または 3FDFH

〈図 2-24〉
タイマの I/O アドレス一覧

〈図 2-25〉 タイマ LSI8253 の周辺回路



〈図 2-26〉 カウンタの割り込み

	PC9801/E/F1,2,3/M2,3	左記以外	動作モード
カウンタ#0	インターバル・タイマ	インターバル・タイマ	モード 0
カウンタ#1	メモリ・リフレッシュ	スピーカ周波数設定	モード 3
カウンタ#2	RS-232-C	RS-232-C	モード 2

このピンが未使用ならば“1”が出力されますが、CLK 入力には 2.4576 MHz が使用されています。(p. 46 コラム参照)

CLK 入力は、拡張スロット用の「システム・クロック」を利用したものが使われます。CPU クロックに依存する機種もありますし、CPU クロックとシステム・クロックが別になっている機種もあり、CPU クロックだけでは判別がつかない場合も多いようです。

■ タイマの用途

98 シリーズでは、3 組のカウンタを持っており、これらは図 2-26 に示すように、インターバル・タイマ、メモリ・リフレッシュ(旧機種)、ビープ(ブザー)用の音源、RS-232-C 用クロック発生に使用されています。

● カウンタ 0(モード#0)インターバル・タイマ

カウンタ 0 はインターバル・タイマで、指定した一定時間後に割り込みをかけることができます。システムは専用に使用していませんので、ユーザがアプリケーションの中で自由に利用できます。

一般的に使用されるモードは「#0」でカウント終了時での割り込みです。指定したカウントが終了したら割り込みがかかるモードで、カウント数を書き込んだ直後からカウントダウンを始め、“0”になったらソフトウェア割り込み(ベクタ#08H)がかかります。割り込みプログラムの中で、8253 に対して再度モード設定すると割り込み解除されます。モード「#0」は 1 回しか割り込みがかからないので、何度も定期的にかけたい場合はモード「#2」を使用します。

インターバル n ms の計算は前述の CLK 入力によって変わります。

▶ システム・クロック 5/10 MHz 時

$n \times 2457.6$ に近い整数(最大 26.666 ms まで可)

▶ システム・クロック 8 MHz 時

$n \times 1996.8$ に近い整数(最大 32.821 ms まで可)

● カウンタ 1(モード#3)ビープ音の周波数設定

カウンタ 1 は PC9801M 以前の機種ではメモリのリフレッシュに使用され、PC9801F では周期が 28.5 μ s に設定されていました。ユーザがこれを変更すると動作が保証されなくなってしまう。

PC9801VM 以降の機種では、ビープ(ブザー)用の音源に使用されています。プログラマブル・カウンタを使用していますから周波数は自由に変えられ、音楽の演奏等も可能です。

カウンタ 1 は機種によって動作は違いますから、プログラムを不用意に触って誤動作するのを防ぐために、ビープ音を設定する場合は、別アドレス(73H \rightarrow 3FDBH, 77H \rightarrow 3FDFH)を設けています。もっとも、現在では PC9801M 以前の機種は、ほとんど使われていませんから問題ないでしょう。

ビープ音の周波数設定ではモード「#3」で方形波レート・ジェネレータを使用しています。これは、なるべくデューティ比 50 % の方形波を作ろうとするモードで、入力値が偶数ならデューティ比 50 % ですが、奇数の場合は、 $n-1/2n$ となり、1 カウント分だけデューティ比 50 % からずれてしまいます。最悪の場合($n=5/n=3$ は禁止)はデューティ比は 2:3 となりますが、可聴領域では「 n 」が大きくなるために、ほぼデューティ比 50 % となります。

ビープ音の周波数(f)設定の計算は以下のとおりです。

$$n = F / f$$

$$F = 2357600 \text{ (Hz)}$$

(システム・クロック 5/10 MHz 時)

$$F = 1996800 \text{ (Hz)}$$

(システム・クロック 8 MHz 時)

ビープ音の ON-OFF には、システム・ポート用の 8255・ポート C・ビット 4 を使用します。

ON: I/O アドレス 37H に 06H を書き込む。

OFF: I/O アドレス 37H に 07H を書き込む。

ビープ回路を図 2-27 に示します。

● カウンタ 2(モード#2)RS-232-C

カウンタ 2 は、RS-232-C 用のクロック発生用で、8251 の TxCLK/RxCLK に接続されています。

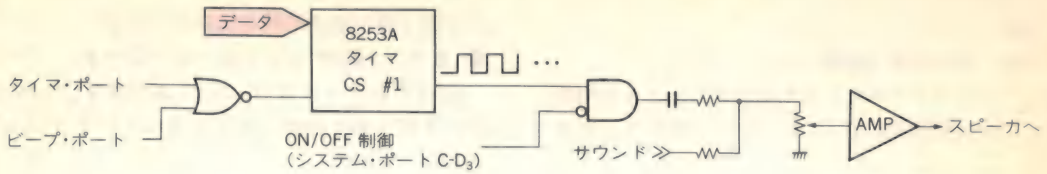
一般によく使用される非同期モードで使用する場合は、ボーレートの 16 倍のクロックが必要になります。CLK 入力に 1.9968 MHz が使用されていると分周比の都合で 9600 bps を超える指定ができなくなります。

RS-232-C 用ボーレート設定は、ビープ音の周波数設定と同じ、モード「#3」で方形波レート・ジェネレータを使用しています。ボーレート算出方法は同期式、調歩同期式 1/16、調歩同期式 1/64 の方式別に図 2-28 にまとめました。

■ LSI の使用方法

8253 にはハードウェア・リセットの端子はありませんし、初期化のためのコマンド等もありません。電源

〈図 2-27〉 ビープ回路



投入以降、何かモードを指定するまでの動作は保証されません。

モードの指定は「コントロール・ワード」への書き込みで行われます。1 バイト (8 ビット) で、カウンタ・チャンネル指定 (0~2)、動作モード指定 (0~5) 等を行います。

チャンネル別にカウンタは三つあり、それぞれアクセス・アドレスも別に割り当てられていますが、コントロール・ワードの書き込みは一つのアドレスしかありません。そのため、どのチャンネルの制御を行うか指定します。

● モード設定 (コントロール・ワード)

8253 のモードは、0~5 まで 6 種類があります。

▶ モード #0

指定した時間が経過すると OUT 端子が “H” (割り込みがかかる) になります。

コントロール・ワードを書くと OUT 端子が “L” になり、カウンタ・データをロードした直後にカウントダウンが始まります。カウントダウンが終了すると OUT 端子は “H” になります。

▶ モード #1

指定された期間 “L” のパルスを出します。

コントロール・ワードを書くと “H” になります。カウンタ・データをロードした時点から、カウント数で指定した期間だけ “L” になります。カウントが終了して、ゲート端子が “L” になれば再びカウントし始めるのですが、PC98 ではゲート端子は使われていないので、モード #0 と同じになります。

▶ モード #2

指定した周期に一度パルス “L” を発生します。

コントロール・ワードを書くと “H” になります。データ・ロードするとカウントが開始され、カウントが終了する直前の 1 パルスだけ “L” になります。終了後は、再び初期値をロードしなおして繰り返されます。カウント中にデータ・ロードしても、次のカウントダウンまで影響を与えません。

▶ モード #3

モード #2 と同様ですが、“H” と “L” の期間が同じ (デューティ比 50 %) になるような出力を出します。ただし、カウンタ・データに “3” は使用できません。

〈図 2-28〉 RS-232-C ボーレート設定

転送速度 (ボー)	同 期 式		調歩同期式 1/16		調歩同期式 1/64	
	5/10 MHz	8 MHz	5/10 MHz	8 MHz	5/10 MHz	8 MHz
19200	128	使用 不可	8	使用 不可	使用 不可	使用 不可
9600	256	208	16	13	使用 不可	使用 不可
4800	512	416	32	26	8	使用 不可
2400	1024	832	64	52	16	13
1200	2048	1664	128	104	32	26
600	4096	3328	256	208	64	52
300	8192	6656	512	416	128	104
150	16384	13312	1024	832	256	208
75	32768	26624	2048	1664	512	416

転送速度 (ボー)	同 期 式	
	5/10 MHz	8 MHz
1200	128	104

▶ モード #4

コントロール・ワードを書くと “H” になります。データ・ロードするとカウントが開始され、カウント終了後の 1 パルスだけ “L” になります。ゲート端子を “L” にするとカウントが停止しますが、PC98 シリーズではゲート端子の操作はできません。

▶ モード #5

モード #4 と似ていますが、ゲート端子を “L” にするとカウントを終了し、再び新しくカウントを再開します。このモードも PC98 シリーズでは意味を持ちません。

98 シリーズでは、8253 のゲート端子はプルアップされているだけで使われていないために、ゲート端子を使用している、モード #1、#4、#5 は意味がありません。一般的に使用されるのは、#0、#2、#3 の 3 種類のようです。

● カウンタ・ラッチ (コントロール・ワード)

カウンタに対して指定する「リード/ライト・モード」というものがあります。これは、カウンタにデータを書いたり、読んだりするモードで、以下のようなものがあります。

- #0: カウント・ラッチ・コマンド
 #1: 下位バイトのリード/ライト
 #2: 上位バイトのリード/ライト
 #3: 下位・上位の順で連続リード/ライト

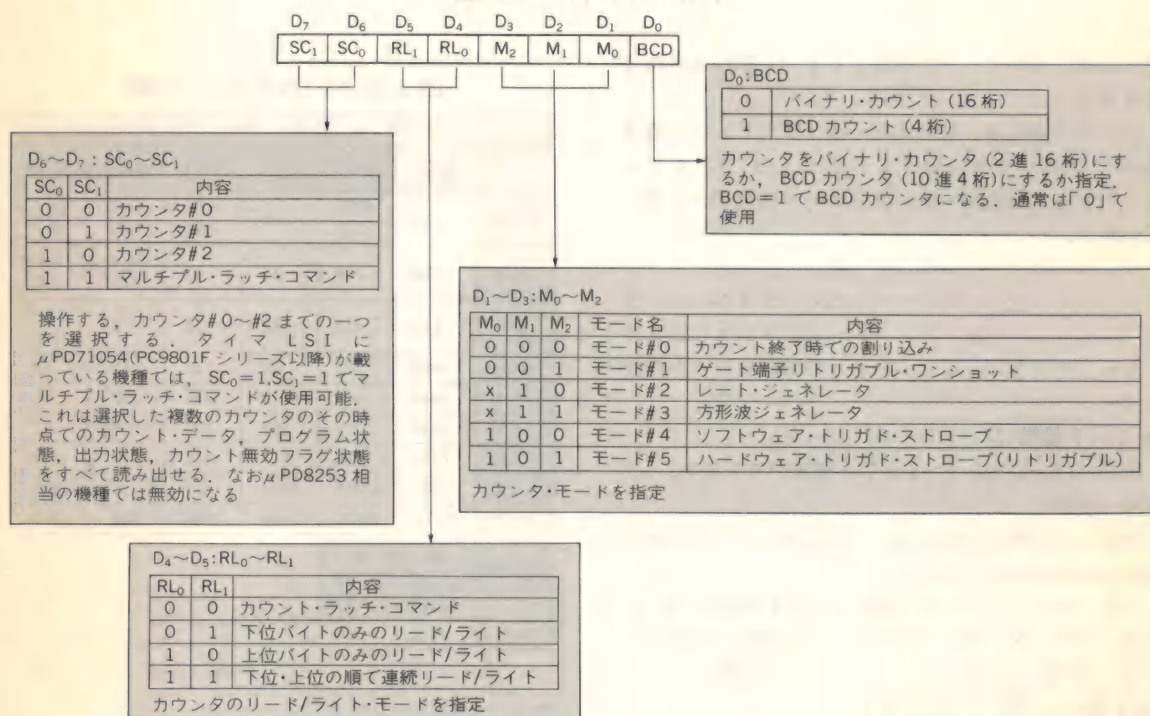
カウント・ラッチ・コマンドを実行すると、その瞬間
 のカウンタの内容がストレージ・レジスタにラッチさ

れ、正確なデータを読み出せます。この間はカウンタ
 自身の動作に影響を与えませんので、インタラプト等
 の正確な途中経過時間を読み出せます。

● カウンタ設定(コントロール・ワード)

8253 のカウンタは 16 ビットあります。しかし、ア
 ドレス割り当てはチャンネルごとに 1 バイト (8 ビット)

〈図 2-29〉 コントロール・ワード

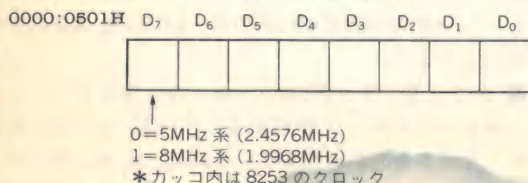


システム共有領域

システム・クロックや、8253 に入力されるクロックの周波数を識別するには、プリンタ・ポートの 42H のビット 5 で認識します。しかし、ハイレゾ・モードやフル・セントロニクス・プリンタ・インターフェース等を使用した場合は、クロック周波数の認識ビットが異なります。

プログラムがハイレゾ・モードでも動作するよう

〈図 2-A〉 システム・クロックの識別



に作るためには、ハイレゾ・モードかどうかを認識し、別のポートでクロック周波数を認識しなくてはなりません。

このクロック周波数は、ハードウェアのポートだけではなく、システム共通領域 (0000:0B01H) でも知ることができます。ビット 7 が "0" ならば、システム・クロックは 5 MHz 系で、8253 に入力されるクロックは 2.4576 MHz となります。"1" の場合は 8 MHz 系で、1.9968 MHz のクロックがかかります。これを利用すると、ノーマル・モード、ハイレゾ・モードの区別なく認識することができます。

ただし、システム共通領域はシステム起動時のクロック周波数を示すもので、起動後にクロックを変更すると正しく認識できなくなってしまいます。

〈図 2-30〉 カレンダ時計のアドレス一覧

命 令	I/O ポート ・ アドレス	READ/ WRITE	デ ー タ								
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
セット・レジスタ	20	W	×	×	DI	CLK	STB	C ₂	C ₁	C ₀	
リード・データ	33	R	×	×	×	×	×	×	×	×	DO

×印：不定

しかありませんので、**上位・下位を別々に読み書きすることになります**。この指定は前述のリード/ライト・モードで行います。

リード/ライト・モードの#1や#2で、**下位または上位のみを設定した場合は、その逆の桁には00Hが入ります**。下位・上位の連続リード/ライトの場合は、連続して読み書きをしなくてはなりません。

◆ コントロール・ワード (77H/3FDFH/出力)

77H でも、3FDFH でも同じポートが読み出せます。これはカウンタ#1の機種による用途の違いで誤動作が起きないようにするために分けられています。

図 2-29 にレジスタを示します。

◆ カウンタ・レジスタ (71H・73H/3FDFH・75H・入出力)

コントロール・ワード設定後に、カウンタのデータを読み書きする場合にアクセスします。

カウンタ#1は、機種によってアドレスが異なり、73H(PC9801M以前)と、3FDBH(PC9801VM以降)があります。これはカウンタ#1の機種による用途の違いで誤動作が起きないようにするために分けられています。

カレンダ時計

▶ 使用 LSI

μPD1990C, μPD4990A

▶ I/O アドレス

20H

33H(システム・ポート)

▶ 使用割り込み

なし

▶ 初期設定命令

なし

図 2-30 にアドレス一覧を示します。

カレンダ時計用の LSI は機種によって異なるものが使用されています。PC9801/E/F/M/U/VF/VM/UV/XA には μPD1990C が使われ、それ以外の機種(PC9801VM 後期モデルを含む)と EPSON の PC シリーズでは μPD4990A が使用されています。

カレンダ時計の LSI には 3 ビットの平行・コマンド・レジスタと、48(40)ビットの時刻データ・レジスタ、4 ビットのシリアル・コマンド・レジスタがあります。CPU からアクセスするには、6 ビット構成のセット・レジスタと、1 ビットのリード・データを使用します。

セット・レジスタは I/O アドレスの 20H 番地で、TTL のラッチにより行われます。**リード・データ**は、システム・ポート用の 8255 のポート B(33H 番地)ビット 0 を使用しています。

平行・コマンド・レジスタは、セット・レジスタのコマンド・コード(3 ビット)にコマンドを書き込み、STB ビットを 0 → 1 → 0 にすることで設定します。

時刻データ・レジスタとシリアル・コマンド・レジスタは、セット・レジスタの DI ビットを使用して、**シリアル形式**で書き込みます。1 ビット書き込むたびに CLK ビットを 1 → 0 にして、これを 48(40)回繰り返して設定します。

日付け・時刻のデータは、年(2 桁)・月(1 桁)・週(1 桁)・日(2 桁)・時(2 桁)・分(2 桁)・秒(2 桁)から構成されます。1 桁は 4 ビットで BCD ですが、月は 01H ~ 0CH になります。

レジスタへの連続書き込みは、1 ~ 5 μs(時間読み出しは 20 ~ 40 μs)と比較的時間がかかります。

◆ μPD 1990 と μPD 4990 の違い

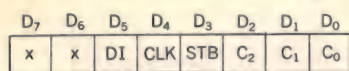
μPD4990A は、μPD1990 の上位互換の LSI でテスト・モード(PC9801 シリーズでは使用されていない)を除き、コンパチブルな動作をします。また、**μPD4990A では「年」を扱うことが可能**になり、「うるう年」の**自動補正**もやってくれます。**BCD データ 2 桁、8 ビット分拡張**されています。

μPD1990 は四つのコマンドを平行で与え、日付けデータはシリアルで送受信します。μPD4990A では、平行・コマンドが八つに拡張され、さらに拡張されたシリアル形式でのコマンドも使え、合計 16 個のコマンドが使用可能になっています。

拡張されたコマンドはタイミング・パルス出力、インターバル出力関係で、インターバル・タイマのような機能を持っているのですが、PC9801 シリーズでは使用されてなく、あまり意味を持ちません。

◆ 曜日の設定

〈図 2-31〉 セット・レジスタ



コマンド・コード
STB と共に用いて μ PD4990 へパラレル・コマンドを与える。
シリアル・コマンド・モード時は、 $C_2, C_1, C_0 = 1, 1, 1$ に保つ。

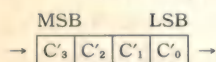
C_2	C_1	C_0	機 能
0	0	0	レジスタ・ホールド
0	0	1	レジスタ・シフト
0	1	0	タイム・セット & カウンタ・ホールド
0	1	1	タイム・リード
1	0	0	TP=64Hz セット
1	0	1	TP=256Hz セット
1	1	0	TP=2048Hz セット
1	1	1	拡張モード・コマンド

μ PD1990 では使用
禁止

ストローブ
 μ PD4990 へコマンドをセットするタイミングを与える
クロック
DO と共に用いて、シリアル・コマンドを μ PD4990 へ時刻
をセットしたり、時刻を読み出すために用いる
入力データ (1 ビットごとに入力される)
CLK と共に用いて μ PD4990 へシリアル・コマンドおよび
時刻データをセットする

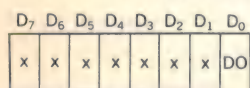
注: パラレル・コマンドで日付け、時刻を書き込むと、年のデータは壊れる

〈図 2-32〉 μ PD4990 シリアル・コマンド



C_3	C_2	C_1	C_0	機 能
0	0	0	0	レジスタ・ホールド
0	0	0	1	レジスタ・シフト
0	0	1	0	タイム・セット/カウンタ・ホールド
0	0	1	1	タイム・リード
0	1	0	0	TP=64 Hz
0	1	0	1	TP=256 Hz
0	1	1	0	TP=2048 Hz
0	1	1	1	TP=4096 Hz
1	0	0	0	TP=1 sec インタラプト出力/カウンタ・リセット
1	0	0	1	TP=10 sec インタラプト出力/カウンタ・リセット
1	0	1	0	TP=30 sec インタラプト出力/カウンタ・リセット
1	0	1	1	TP=60 sec インタラプト出力/カウンタ・リセット
1	1	0	0	インタラプト出力リセット
1	1	0	1	インタラプト・タイマ・スタート
1	1	1	0	インタラプト・タイマ・ストップ
1	1	1	1	テスト・モード・セット

〈図 2-33〉 リード・レータ



出力データ (1 ビットごとに出力
される)
CLK と共に用いて μ PD4990 から
時刻を読み出す。

MS-DOS のシステム・コールの「日付けの取得」は、曜日情報をカレンダー時計から得ずに、年月日の情報から自分で計算します。また、「日付けの設定」でも、カレンダー時計に曜日情報を正確に書き込まずに、常に OOH (日曜日) と設定してしまいます。

BIOS を使用した場合は、カレンダー時計が保持している曜日情報を、正確に入出力しますので、MS-DOS で取得した曜日情報と食い違う可能性があります。曜日情報を有効にする場合は、MS-DOS のシステム・コールを使用したほうが安全です。

◆ セット・レジスタ (20H/ライト)

$D_0 \sim D_5$ までのデータが、カレンダー時計 IC にラッチを通して接続されています。

▶ $D_0 \sim D_2$: $C_0 \sim C_2$ (コマンド・コード)

コマンド入力、ファンクション・モードの選択等を指示します。

▶ D_3 : STB (ストローブ入力)

ストローブ入力、データ・コマンドの入力時に使用する書き込みストローブ信号です。

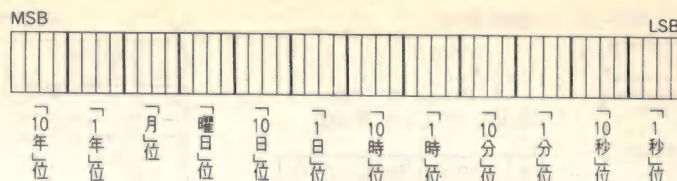
コマンド入力を確定してから $2 \mu s$ 後に、ストローブ入力を $0 \rightarrow 1$ に変化させ、 $5 \mu s$ ($40 \mu s$: Time Read コマンド時のみ) 後に、 $1 \rightarrow 0$ にします。コマンド入力されたデータは、その後 $2 \mu s$ 間変更してはなりません。

▶ D_4 : CLK (クロック入力)

カレンダー時計 IC 内部のシフトレジスタのシフト・クロック入力です。

クロック入力の立ち上がり ($0 \rightarrow 1$) でデータ (DI) を読み込みシフトします。クロック入力の前後の $2 \mu s$

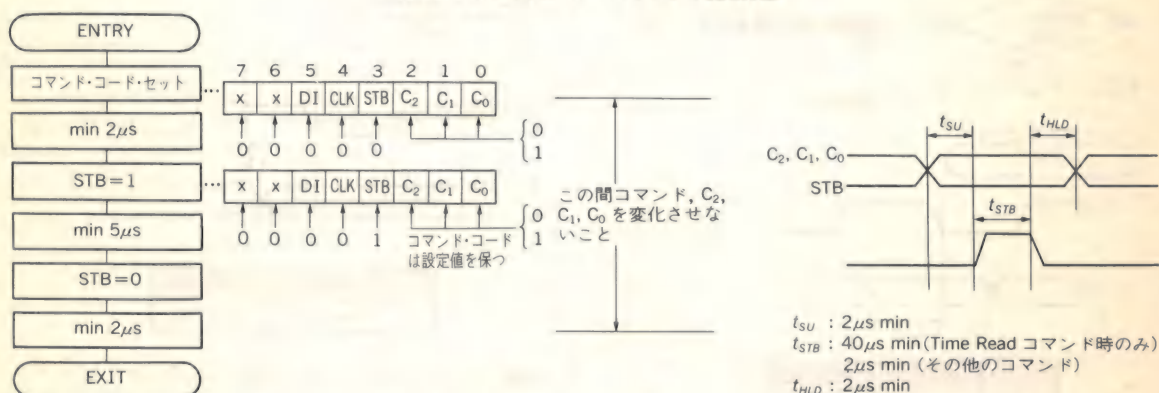
〈図 2-34〉
入出力データ形式



曜日	
日曜	0000
月曜	0001
火曜	0010
水曜	0011
木曜	0100
金曜	0101
土曜	0110

項目	形式	範囲
月	HEX DECIMAL	01H~0CH
曜	BCD	00H~06H
日	BCD	01H~31H
時	BCD	00H~23H
分	BCD	00H~59H
秒	BCD	00H~59H

〈図 2-35 (a)〉 パラレル・コマンドの使用法



(4990 は 1 μ s) 間は DI を変化させてはなりません。

データ出力 (DO) へ時刻データをシリアルで出力させるときにも使用されます。

▶ D_5 : DI (データ入力)

シリアル・コマンドの入力や、時刻データの入力を行います。CLK 信号とタイミングをとってシリアルで入力します。

図 2-31 にセット・レジスタを、図 2-32 に μ PD4990 のシリアル・コマンドを示します。

◆ リード・データ (33H/リード)

▶ D_0 : DO (データ出力)

CLK とタイミングをとって、時刻データの出力に使用します。

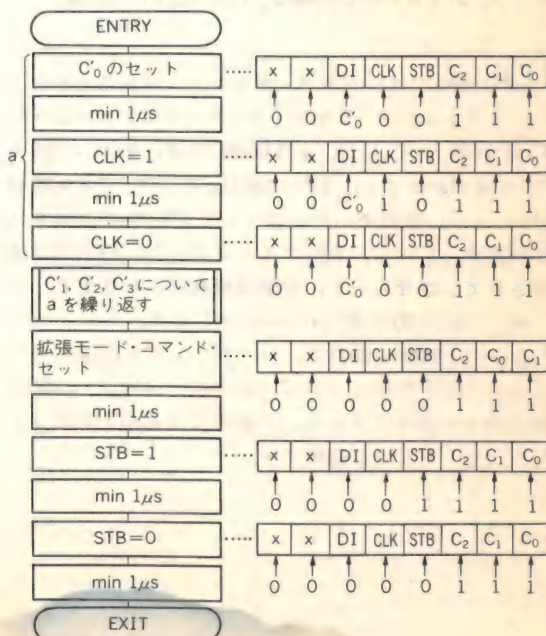
このポートはシステム・ポートの 8255 のポート B を使用しています。

図 2-33 にリード・データを図 2-34 (a), (b) に入出力データ形式を示します。

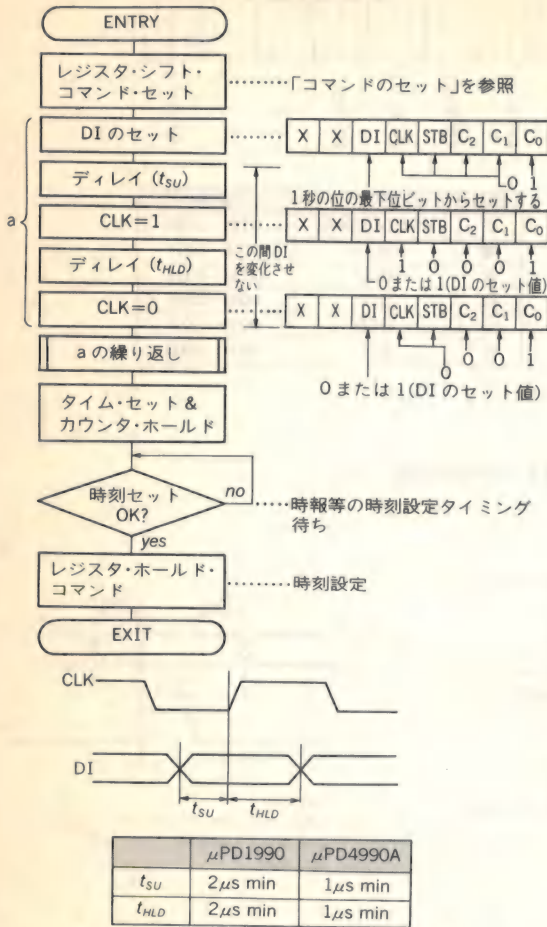
◆ カレンダー時計 IC のコマンド・コード

セット・レジスタの $C_0 \sim C_2$ に設定するデータです。

〈図 2-35 (b)〉 シリアル・コマンドの使用法



〈図 2-35 (c)〉 時刻の設定



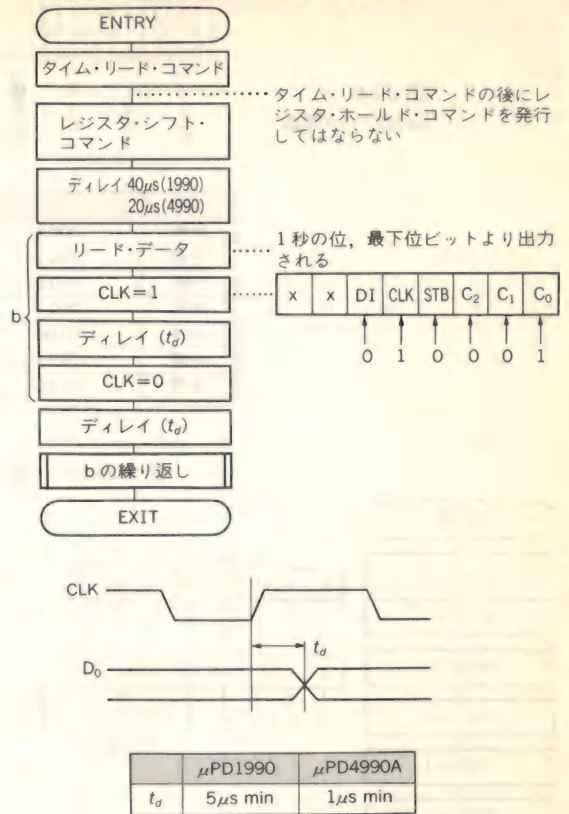
注: シリアル・コマンド・モード時は, $C_2, C_1, C_0 = 1, 1, 1$ に保つ

μPD1990 ではレジスタ・ホールド, レジスタ・シフト, タイム・セット&カウンタ・ホールド, タイム・リードの四つですが, μPD4990 では, それに加え TP=64 Hz セット, TP=256 Hz セット, TP=2048 Hz セット, 拡張モード・コマンドの四つが追加されています。しかし, TP(タイミング・パルス)出力は使用されていないので, 設定は無意味になります。

使用方法を図 2-35 (a)~(d) に示します。

カレンダ時計の操作は, ハードウェアを直接利用した場合の動作保証はされていません。日付け・時間の読み書きが基本ですから, 一般的には BIOS を使って行ったほうが安全で確実です。

〈図 2-35 (d)〉 時刻の読み出し



注意: シリアル・コマンド・モード時は, $C_2, C_1, C_0 = 1, 1, 1$ に保つ

システム・ポート

▶ 使用 LSI

μPD8255A 相当

▶ I/O アドレス

31H, 33H, 35H, 37H

▶ 使用割り込み

なし

▶ 初期設定命令

8255=92H

システム・ポートでは, ビープの ON/OFF や, カレンダ時計の読み出し, ディップ・スイッチの読み出し等を行うことができ, LSI にプログラマブル・パラレル・ポートの μPD8255A の相当品(以下, 8255 と略す)を使用しています。

システム・ポートでの 8255 は, ポート A がモード 0/入力, ポート B がモード 0/入力, ポート C がモード 0/出力に初期設定されます。初期設定命令は 92H です。I/O アドレスの一覧を図 2-36 に, システム・

〈図 2-36〉 システム・ポートの I/O アドレス一覧

命 令	I/Oポート・アドレス	READ/WRITE	デ ー タ								備 考
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
ライト・モード	37	W	1	0	0	1	0	0	1	0	8255 のモード・セット
ライト・ポート C	37	W	0	0	0	0	0	0	0	0/1	RXRE F/F の ON/OFF 0 : OFF, 1 : ON
	37	W	0	0	0	0	0	0	1	0/1	TXEE F/F の ON/OFF 0 : OFF, 1 : ON
	37	W	0	0	0	0	0	1	0	0/1	TXRE F/F の ON/OFF 0 : OFF, 1 : ON
	37	W	0	0	0	0	0	1	1	0/1	スピーカ F/F の ON/OFF 0 : ON, 1 : OFF
	37	W	0	0	0	0	1	0	0	0/1	メモリ・チェック Enable の ON/OFF 0 : Disable, 1 : Enable
	37	W	0	0	0	0	1	0	1	0/1	SHUT ₁ の ON/OFF
	37	W	0	0	0	0	1	1	0	0/1	プリンタの PSTB 信号 マスク F/F の ON/OFF 0 : イネーブル 1 : マスク
	37	W	0	0	0	0	1	1	1	0/1	SHUT ₀ の ON/OFF
ライト・ポート C	35	W	SHUT ₀ (* 1)	PSTBM (* 2)	SHUT ₁ (* 1)	MCKEN (* 3)	BUZ	TxRE	TxEE	RxRE	PORT C の信号は本命令でも ON/OFF 可
リード・ポート C (診断用)	35	R	SHUT ₀ (* 1)	PSTBM (* 2)	SHUT ₁ (* 1)	MCKEN (* 3)	BUZ	TxRE	TxEE	RxRE	PORT C の状態を読み取る
リード・ポート B	33	R	\overline{CI}	\overline{CS}	\overline{CD}	INT _s	CRTT	IMCK	EMCK	CDAT	PORT B を通して各種信号を読み取る
リード・ポート A	31	R	\overline{SW}_8	\overline{SW}_7	\overline{SW}_6	\overline{SW}_5	\overline{SW}_4	\overline{SW}_3	\overline{SW}_2	\overline{SW}_1	PORT A を通して各種スイッチ信号を読み取る

* 1 : SHUT₀, SHUT₁ は 80286/386/486/Pentium 搭載機種のみ

* 2 : プリンタ PST 信号マスク F/F (PSTBM) は、PC9801 では 94H ポートの D₄ ビットを使用

* 3 : メモリ・チェック Enable (MCKEN) は、PC9801U2 では未使用

ポートの周辺回路を図 2-37 に示します。

◆ ポート A (31H/リード)

ポート A は入力に設定されていて、**ディップ・スイッチ SW₂の内容を読み出す**ことができます。それぞれのビットが意味する内容は機種によって若干変わります。ディップ・スイッチは、ハードウェアのものと、メニュー画面から設定するソフトウェアのもの、ハードウェアとソフトウェアの混在したものがあります。ポート A 入力を図 2-38 に示します。

ディップ・スイッチは、**ON** にすると対応するポート A のビットが **"0"** になり、**OFF** になると **"1"** になります。

◆ ポート B (33H/リード)

ポート B は入力に設定されていて、主に、RS-232-C の信号読み取り、カレンダー時計の読み出し、RAM

のパリティ・エラーの原因表示等の読み出しができます。ポート B 入力を図 2-39 に示します。

▶ D₀ :

カレンダー時計からのデータをシリアル・データとして読み出すポートです。

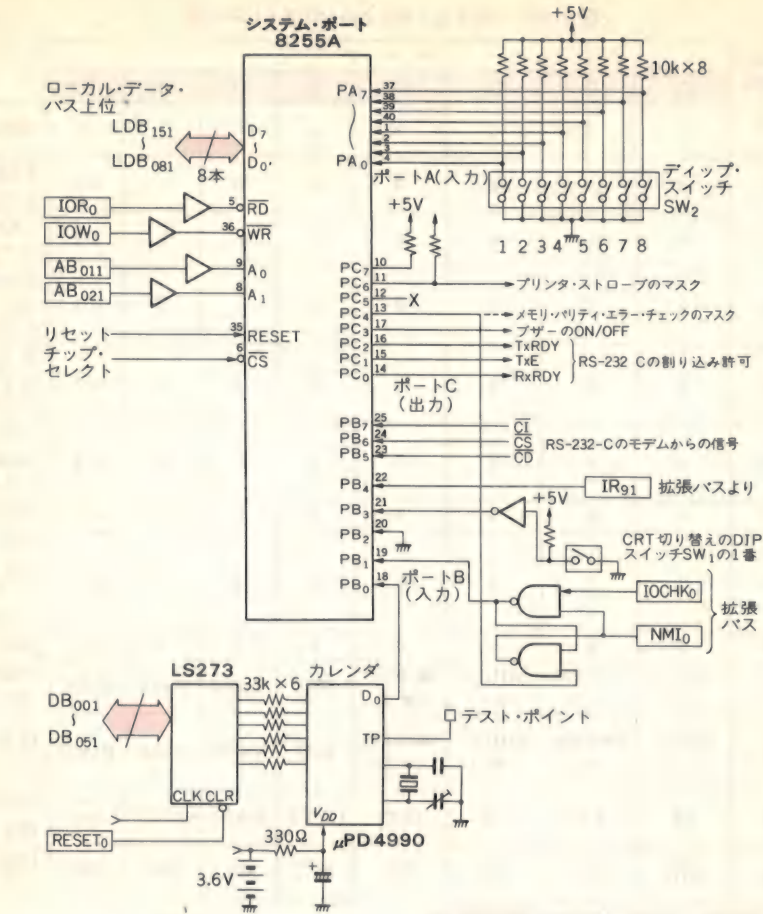
▶ D₁ : 外部 RAM パリティ・エラー/D₂ : 内部 RAM パリティ・エラー

RAM の**パリティ・エラー**が起きたときの**原因区別**用です。D₂は内蔵 RAM、専用増設 RAM 用で機種によっては未使用なものもあります。D₁は「拡張 RAM 用」です。“0”でエラーなし、“1”でエラー発生です。RAM のパリティ・エラーが発生すると、ハードウェア割り込みである NMI 割り込み(ベクタ #2)がかかります。この割り込みプログラムはエラー原因を知るために使います。

▶ D₃ : CRT タイプ

1 = 高解像度, 0 = 標準解像度を選びます。ディッ

<図 2-37>
 システム・ポート
 とカレンダー



<図 2-38> ポート A 入力

データ・ビット	信号名	備 考
D ₇	SW ₈	ON : GDC 5 MHz, OFF : GDC 2.5 MHz (* 1)
D ₆	SW ₇	未使用
D ₅	SW ₆	ON : 固定ディスク切り離し, OFF : 接続 (* 2)
D ₄	SW ₅	ON : メモリ・スイッチ保持, OFF : メモリ・スイッチ初期化
D ₃	SW ₄	ON : 25 行/画面, OFF : 20 行/画面
D ₂	SW ₃	ON : 80 文字/行, OFF : 40 文字/行
D ₁	SW ₂	ON : ターミナル・モード, OFF : BASIC
D ₀	SW ₁	常に OFF

* 1 : PC9801/E/F1,2,3/M2,3/U2/VF2/VM0,2,4 では未使用
 * 2 : 固定ディスク内蔵モデルのみ, その他では未使用
 * 3 : PC9801N ではディップ・スイッチが 1 個となったため, D₄, D₇ビットのみ有意となる

プ・スイッチの SW₁₋₁が読み取れます。
 ▶ D₄ : INT₃
 ハード・ディスク割り込み信号です。
 ▶ D₅ : CD/D₆ : CS(CTS)/D₇ : CI
 RS-232-C 用に使用されている 8251 では読み取る
 ことのできない, CI, (CTS)CS, CD, の信号を読み
 取ることができます。RS-232-C の各信号が, +V

(OFF)の時や, 開放されている時に対応するビット
 が“1”になります。-V(ON)の時は“0”になりま
 す。
 ■ ポート C(35H/ライト)
 ポート C は出力に設定されていて, RS-232-C の割
 り込み制御, ビープの ON/OFF, プリンタの PSTB

〈図 2-39〉 ポート B 入力

データ・ビット	信 号 名	備 考
D ₇	CI (RS-232-C)	RS-232-C CI 信号
D ₆	CS (RS-232-C)	RS-232-C CS 信号
D ₅	CD (RS-232-C)	RS-232-C CD 信号
D ₄	INT ₃	固定ディスク INT 信号
D ₃	CRT TYPE	1: 高解像度, 0: 標準解像度
D ₂	内部 RAM パリティ・エラー	標準 RAM のパリティ・エラー
D ₁	外部 RAM パリティ・エラー	拡張 RAM のパリティ・エラー
D ₀	カレンダー時計の読み出しデータ	

〈図 2-40〉 ポート C 出力

データ・ビット	信 号 名	備 考
D ₇	SHUT ₀	
D ₆	PSTB マスク	プリンタの PSTB 信号のマスク
D ₅	SHUT ₁	
D ₄	メモリ・チェック Enable	1: エラー登録する*, 0: エラー登録しない
D ₃	ブザー制御 F/F	1: ブザー停止, 0: 鳴動
D ₂	TXR Enable F/F	RS-232-C の TXRDY による割り込みの Enable
D ₁	TXE Enable F/F	RS-232-C の TXEMPTY による割り込みの Enable
D ₀	RXR Enable F/F	RS-232-C の RXRDY による割り込みの Enable

信号の制御等に使われています。このポート C はバイト単位(8 ビット 1 組)でなく、ビット単位で制御できる「ビット・モード」を使って操作するのが便利です。ポート C 出力を図 2-40 に示します。

▶ D₀: RxRDY 割り込みイネーブル/D₁: TxEMPT 割り込みイネーブル/D₂: TxRDY 割り込みイネーブル

RS-232-C 用割り込みのマスク用です。「1」で割り込み禁止、「0」で割り込み可能になります。

▶ D₃: ビープの ON-OFF

ビープ音の ON-OFF 用です。「0」で鳴り(ON)、「1」で停止(OFF)します。

▶ D₄: パリティ・エラーのイネーブル

パリティ・エラー用の NMI 割り込みの制御で、「0」で禁止、「1」で許可です。

▶ D₆: PSTB マスク

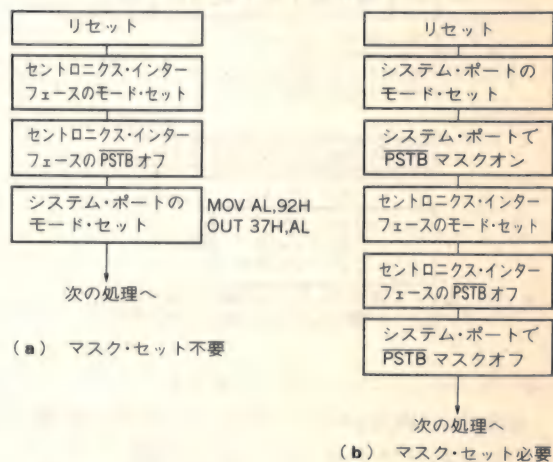
プリンタ・ストローブ信号のマスク制御用です。8255 は出力モード設定直後はすべて「0」となるために、プリンタ用 8255 とシステム・ポート用 8255 を設定する際に設定順序を考えないと、プリンタが誤動作する可能性があります。モード設定フローを図 2-41 に示します。

▶ D₅: SHUT₁/D₇: SHUT₀

80286 以上の CPU を使用する際に、ハードウェア・リセットがかかったか、プロテクト・モードからの復帰なのかを判別するポートです。CPU に 8086 や V30 を使用した機種では未使用です。

80286 では、プロテクト・モードから、リアル・モー

〈図 2-41〉 モード設定



ドへ復帰する際に、CPU に対してリセットをかけなくてはならないため、ソフトウェアから CPU のみをリセットすることができます。このポートを読むことで、ハードウェア・リセットがかかったのか、CPU だけリセットされたのか判別することができます。

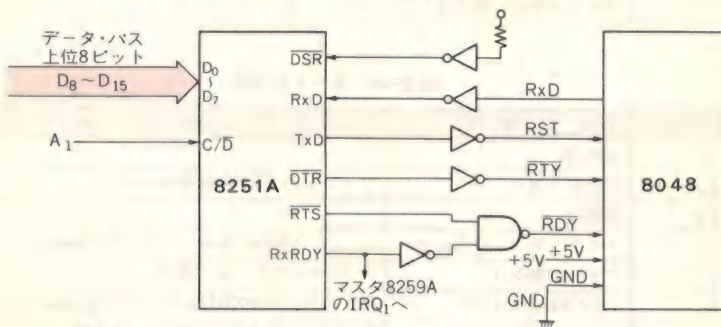
仕組みは、ハードウェア・リセットがかかると、8255 も初期化され、ポート C は入力モードになり、ビット 5, 7 の入力をプルアップしておけば「1」が読み出せます。その後、プログラムが 8255 のポート C を「出力」として設定すれば、ビット 5, 7 は「0」になりますので区別ができます。

SHUT₀=0: プロテクト・モードから復帰

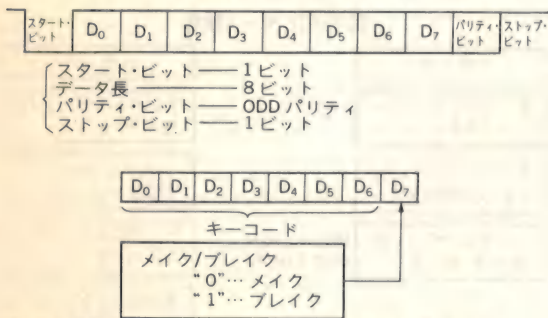
〈図 2-42〉 キーボード・インターフェースの I/O アドレス

命 令	I/O ポート ・ アドレス	READ/ WRITE	機 能
モード/コマンド・ライト	43	W	モード・セット コマンド・セット
データ・リード	41	R	μPD8251A にロードされた 1 バイト・データを読み出す
ステータス・リード	43	R	μPD8251A のステータスを読み出す

〈図 2-44〉 キーボード・インターフェースの回路



〈図 2-43〉 シリアル・データのフォーマット(上),
データのフォーマット



注:メイクはキーが押下されたときの割り込みを示し,
ブレイクはキーが離されたときの割り込みを示す

SHUT₀=1:ハードウェア・リセット

起動時に BIOS がポート 35H から SHUT₀を読み、0 であればプロテクト・モードからの復帰(ソフトウェア・リセット)と判別し、SP ← 0000 : 0404H/SS ← 0000 : 0406H として「FAR RET」します。1 であれば、電源投入時カリセット・スイッチが押された(ハードウェア・リセット)と判別します。

例外として、EPSON の PC286 では、プロテクト・モードからの復帰判別に、OC03H のポートを使用します。読み出したデータが 50H (大文字英字の「P」) であれば、プロテクト・モードからの復帰と判別し、SP ← 0000 : 0400H / SS ← 0000 : 0402H として「FAR RET」します。

キーボード・インターフェース

►使用 LSI

μ PD8251A 相当(本体側)

μPD8048 相当(キーボード側)

▶ I/O アドレス

41H, 43H

▶使用割り込み

IR₁(ベクタ# 09H)

▶ 初期設定命令

$$8251 = 5EH$$

図 2-42 に I/O アドレス一覧を示します。

キーボードには、セパレート・タイプ(本体とケーブルにより接続される)と、ノート・パソコンのように本体と一体型のものがあります。さらに、「CAPS」、「カナ」のキーがメカニカル・ロックされるものと、ソフトウェア制御によりLED等で表示されるものの2種類があります。

キーの数は、79, 84, 100, 101, 106, 107の6種類があります。98LTは79個、それ以外のノート・ラップトップでは84個、CPUに8086を持つ古い機種は100個、CPUにV30や80286を持つ古い機種では「NFER」キーが増えて101個、比較的新しいものはファンクション・キー「vf・1～vf・5」が5個増えて106個で最近の標準となっています。ハイレゾ・モードを持つタイプでは「HOME」キーが独立して107個あります。

8255について

8255 は、8 ビットの汎用パラレル入出力ポートを 3 組(合計 24 ビット)持った LSI です。ポートは、「ポート A」、「ポート B」、「ポート C(上位 4 ビット)」、「ポート C(下位ビット)」の四つに分けられ、プログラムから自由に「入力」、「出力」へ設定できます。また、「ポート C」は、ビット単位で内容を可変できる「ビット操作モード」を持ちます。

モード設定では、「ポート A」、「ポート C(上位4ビット)」をAグループ、「ポート B」、「ポート C(下位ビット)」をBグループと区別します。

I/O アドレスは、「ポート A, ポート B, ポート C」と「コントロール・レジスタ」の四つがあります。コントロール・レジスタで、各ポートの「入力/出力の指定」と「モード設定」、そして「ビット操作モード」を行います。

● モード#0

8255 には三つのモードがあります。モード 0 は、単純な入出力モードで、三つのポートのすべてが汎用入出力ポートになります。出力設定では CPU がライトしたデータがそのままポートに現れ、CPU がリードすると前にライトしたデータが読めます。入力設定にした場合は、ポートの内容をそのまま CPU が読み取ることができます。PC9801 シリーズの「ノーマル・モード」では、すべてモード 0 で使われています。

● モード#1

「入力」、「出力」の片方向のハンドシェイクができます。ハンドシェイクとは、CPUがポートを読み出すまで、ハード的に相手からのデータ送出を停止させられる機能で、同時に割り込み処理も可能になります。プリンタ等のCPUに対して著しく速度が違う周辺機器を制御する場合に、割り込み等での処理が可能になり便利です。ハイレゾ・モードでのプリンタ出力に使われています。

モード1では、ポートA/Bを使用することができ、ポートCはハンドシェイクを実現させるために、1組で3個(A/B両方なら6個)を制御線に使用します。残ったポートCは、モード0の汎用ポートとして使用することができます。

● モード#2

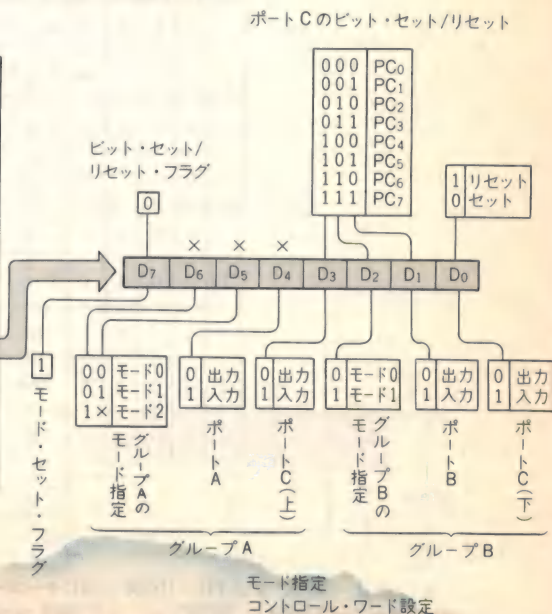
入出力の双方向ハンドシェイクができるモードですが、ポート A だけしか使用できません。この場合はポート C の 5 個が制御線に使用されます。PC9801 シリーズでは使用されないモードです。

● 8255 の初期設定

図 2-B はコントロール・レジスタの設定方法です。最上位ビットが「1」の場合は、8255 の初期設定になり、いつでも自由に設定が可能です。最上位ビットが「0」の場合は、「ビット操作モード」になります。ポート C は常にモード 0 に設定されますが、グループ A/B で、モード 1/2 に設定する場合は、その一部が制御線として使われます。

〈図 2-B〉 8255A のコントロール・ワードと設定例

IN命令で、A, B, Cの各ポートのデータを読み取れる		CS	A ₁	A ₀	WR	RD	入力動作 (READ)
		L	L	L	H	L	ポートA→データ・バス
		L	L	H	H	L	ポートB→データ・バス
		L	H	L	H	L	ポートC→データ・バス
OUT命令で、A, B, Cの各ポートへデータを出力できる							出力動作 (WRITE)
		L	L	L	L	H	データ・バス→ポートA
		L	L	H	L	H	データ・バス→ポートB
		L	H	L	L	H	データ・バス→ポートC
モード設定のためのコマンドを書き込む。このモード設定で各ポートの状態が決まる。D ₇ =0のときは、ポートCの各ビットのON/OFFの制御ができる							コントロール
		L	H	H	L	H	データ・バス→コントロール・レジスタ
							機能なし
		H	x	x	x	x	データ・バス→3ステート
		L	H	H	H	L	イリーガル状態
		L	x	x	H	H	データ・バス→3ステート



キーボードと本体とのデータのやりとり

キーボードと本体との間は、シリアル方式でデータ転送されています。4本の制御線・データ線と、+5Vの電源ライン、グラウンドの合計6本でつながれており、シリアル通信にはRS-232-Cと同じμPD8251A相当(以下8251と略す)が使用されています。キーボード側はμPD8048相当のワンチップCPUで制御されています。

シリアル通信方式は、19200 bps、8ビット、スター

ト・ビット1、ストップ・ビット1、パリティ奇数、調歩同期式、TTLレベルになっています。データの発生は、キーボードが押されたとき(Make)と離れたとき(Break)で、0.5秒以上押してリピート機能が動作しているときは、MakeとBreakが連続してキーボード側で発生されます。

シリアル・データのフォーマットおよびデータのフォーマットは、図2-43のようになります。キーボード(8048)から送られてくるデータは8ビット構成ですが、最上位ビット(D₇)は、Make/Breakの情報を表

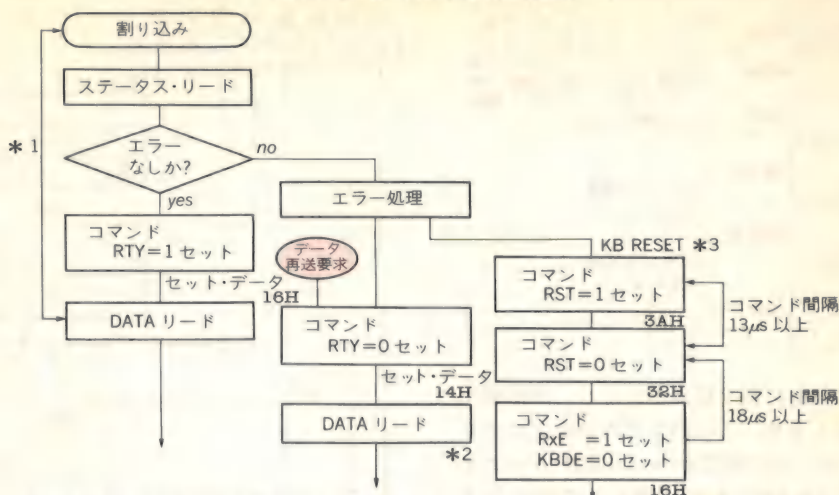
<図2-45> キー・コード一覧

Make															
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	0	0	0	0	0	0
								0	0	0	0	1	1	1	1
								0	0	1	1	0	0	1	1
								0	1	0	1	0	1	0	1
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	ESC	Q	タ	F	ハ	<	ネ	STOP
0	0	0	1	1	1	1	1	!	W	テ	G	キ	>	ル	NER
0	0	1	0	0	0	0	0	"	E	イ	H	ク	?	メ	f.11
0	0	1	1	1	1	1	1	#	3	ア	R	ス	J	マ	f.12
0	1	0	0	0	0	0	0	\$	4	ウ	T	カ	K	ノ	f.13
0	1	0	1	1	1	1	1	%	5	エ	Y	ン	L	リ	f.14
0	1	1	0	0	0	0	0	&	6	オ	U	ナ	;	レ	f.15
0	1	1	1	1	1	1	1	'	7	ヤ	I	ニ	:	ケ	f.6
1	0	0	0	0	0	0	0	(8	ユ	O	ラ)	ム	f.7
1	0	0	1	1	1	1	1)	9	ヨ	P	セ	Z	ツ	f.8
1	0	1	0	0	0	0	0	A	0	フ	@	ッ	X	サ	f.9
1	0	1	1	1	1	1	1	-	B	=	[リ	C	ソ	f.10
1	1	0	0	0	0	0	0	^	C	へ	←	V	ヒ	→	f.3
1	1	0	1	1	1	1	1	¥	D	ー	A	チ	B	コ	=
1	1	1	0	0	0	0	0	BS	E	ト	S	ト	N	ミ	HOME
1	1	1	1	1	1	1	1	TAB	F	シ	D	シ	M	モ	HELP
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	8	9	A	B	C	D	E	F
								0	1	0	1	0	1	0	1
								0	0	1	1	0	0	1	1
								0	0	0	0	1	1	1	1
								1	1	1	1	1	1	1	1

注: [vf.1] ~ [vf.5] は [f.11] ~ [f.15] と同じキーコードを発生する

キーコード 5EH の [HOME] キーは PC98XA model 1, 2, 3/11, 21, 31/XL model 1, 2, 4/XL²/RL のみにある

〈図 2-46〉 キーボード割り込み処理



- * 1: 割り込みから、DATA リードまでの時間が $37\mu\text{s}$ 以上になる必要がある (KB に対して $\text{DRY}=1$ のパルス幅が $37\mu\text{s}$ 以上必要となる)
 * 2: ステータスに異常があり、RTY=0 をセットした場合でも、必ず DATA を引き取ること
 * 3: 数回リトライを行ってもステータスに異常がある場合には、KB RESET を行う

し、キーコードは7ビットになっています。

制御線は、RxD(RxD/括弧内は 8251 の信号線名)でキーボードからのデータを受けます。RDY(RxRDY&RTS)はキーボードへデータ受け取り準備完了を示します。RTY(DTR)はキーボードへ前回受けたデータの再送信要求信号です。RST(TxD)はキーボードのCPUを初期化する信号です。8251のTxDは本来はデータ送信用に使用している端子なのですが、ブレーク信号発生コマンドを実行すると、TxDが“L”レベルになることを利用して、汎用出力端子として使用しています。

図 2-44 にキーボード・インターフェースの回路を示します。

◆ キーボードのソフトウェア制御

ソフトウェア制御の可能なキーボード(主に PC9801RA 以降)では、本体側からキーボードの制御ができ、「キーボードのリセット」、「キーボードのタイプ認識」、「LED の制御」等が可能です。また、「CAPS」「カナ」の状態は本体の電源を消しても記憶されています。

ソフトウェア制御は、RST(TxD)用の制御線を使ってコマンド(データ)を送ります。本来は初期化用の制御線ですから、コマンド発生にはいくつかの条件があります。RST 信号発生直後(0 ms)~3 ms と、10~50 ms の間は、コマンド発生禁止です。また、キーボードから本体側へデータを送っている途中は、データをすべて送り終わるまでコマンドは処理されません。

ソフトウェア制御の可能なキーボードかどうかの判別は、システム共通領域の 0000:0481H のビット 6

で調べられます。“1”の場合がソフトウェア制御の可能なキーボードです。

古いタイプのキーボードでは、本体からキーボード・ケーブルを抜いて、再度差ししても、キーボードのCPUがリセットされずに、本体をリセットするまで使用不能でしたが、ソフトウェア制御ができるキーボードでは、脱着してもキーボードが使用できなくなることはありません。EPSON-PC シリーズでは、メカニカル・ロックのキーボードを脱着しても使用不能にはなりません。

ソフトウェア制御のキーボードを古いタイプのキーボードが付く機種につなげたり、その反対をしてもキーボードとして動作します。しかし、電源を切ると、キーボードの状態と本体のキーの記憶状況(古いタイプの機種では記憶されない)とが合わなくなる場合があります。電源を入れるたびに「CAPS」、「カナ」のキーを空押しすれば、状態は一致できます。

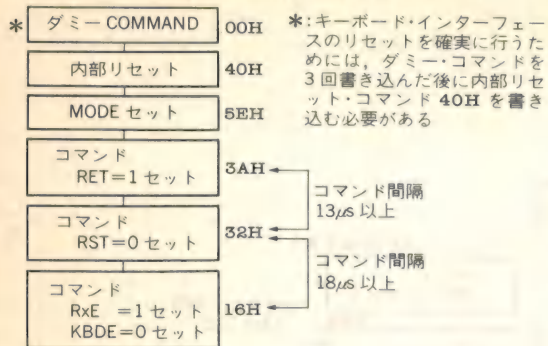
◆ キーボード割り込みの処理

キーボードから送られるシリアル・データ(キー・コード)は、図 2-45 のように 107 個(SHIFT キーとリターン・キーが二つずつあるので 105 種類)のキーを 7 ビットで表します。そのキーが押されたとき(Make)に(0)、離されたときに(Break)に(1)のデータを 1 ビットで表し、合計 8 ビットのデータになります。

ここで発生するデータは、JIS コードとは無関係で、キーボードのキーそのものに与えられたコード番号になっていて、SHIFT キーやコントロール・キー等もデータとして得られます。

8251 がキー・コードを受信すると、割り込みコン

〈図 2-47〉 8251 の初期設定



ローラ (8259) の IRQ₁ に割り込みがかかり、内部割り込み 09H が発生します。この割り込み処理プログラムは、押されたキーが COPY キーか STOP キーであれば、内部割り込み 05H または 06H を発生させます。受信したキー・コードからキー入力状態テーブル (0000 : 052AH~0539H) を作成し、この内容からシフト状態を考慮したうえで JIS コードへコード変換します。

変換された JIS コードと、元のキー・コードを合わせた2バイトを、0000 : 0502H~0521H の BIOS のキー・バッファへ格納し、同時に各ポインタを書き換え、処理は終了します。

EPSON の PC シリーズの一部の PC の機種では、キーボード割り込みの処理が、NEC の 98 シリーズと違うものもありますので、注意が必要です。

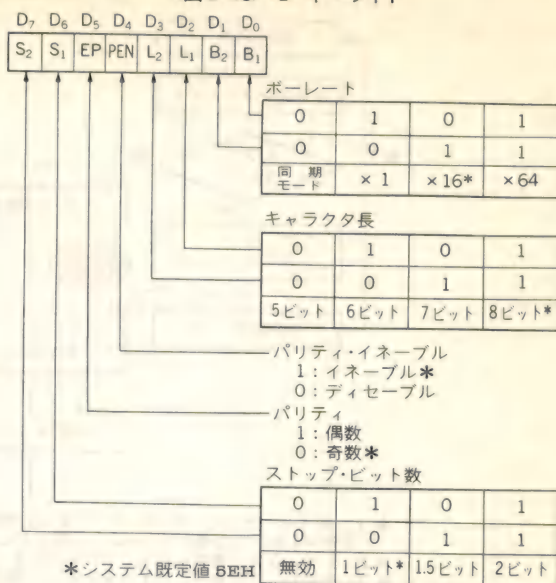
図 2-46 にキーボード割り込み処理フローを示します。

◆ キーボード・インターフェースの初期化と制御

キーボード・インターフェース用の 8251 の初期設定データは 5EH です。8251 の初期設定後は、キーボードをリセットするために「ブ레이크信号」を送り、キーボード送信許可 (RTS=0)、再送信要求有効 (DTR=0)、受信割り込み許可 (Rx E=1)、送信禁止 (Tx EN=0)、エラー・リセット (ER=1) にして、キーボードからのデータを待ちます。

8251 の I/O アドレスは二つあり、モード・コマンド書き込み (設定)/ステータス読み出しポートと、データの読み書きポートです。モード・コマンド設定ポートは一つのアドレスを共用しており、モード設定 (初期化データ 5EH を書き込む) は 8251 をリセットした直後に一度だけ設定でき、二度目からはコマンド設定ポートになります。8251 のリセットはコマンド設定 (ソフトウェア) から行うので、決められた手順どおりに「リセット→モード設定 (図 2-47)」する必要があります。

〈図 2-48〉 モード・ライト



◆ 8251 のレジスタ

8251 の各レジスタの内容は以下のとおりです。

● モード設定 (43H/ライト)

キーボード・インターフェースでは、19200 bps、8ビット、スタート・ビット 1、ストップ・ビット 1、パリティ奇数、調歩同期式に固定されています。8251 の Tx C/Rx C には、307.2 kHz が供給されているのでボーレート指定は「×16」となり、初期設定データは 5EH となります (図 2-48)。

● コマンド設定 (43H/ライト)

図 2-49 はコマンドの設定です。

▶ D₀ : Tx EN

送信の許可/禁止を指示します。送信禁止 (Tx EN=0) にすると、その時点で書き込まれているデータをすべて送出してから送信を停止します。

▶ D₁ : RTY (DTR)

受信データにエラーがあった場合に、再送信要求を指示します。RTY=0 で再送信要求です。8251 の汎用出力ポート (DTR) の制御用です。

▶ D₂ : Rx EN

受信の許可/禁止を指示します。Rx EN=0 で受信禁止です。

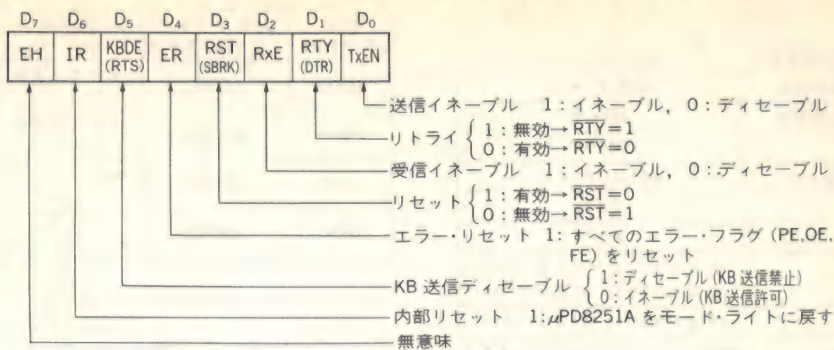
▶ D₃ : SBRK

キーボードのリセットを行います。リセット時には 13 μ s だけ SBRK=1 とします。本来ブ레이크信号送出用で、SBRK=1 のときに 8251 の Tx D 出力を“L”レベルにします。

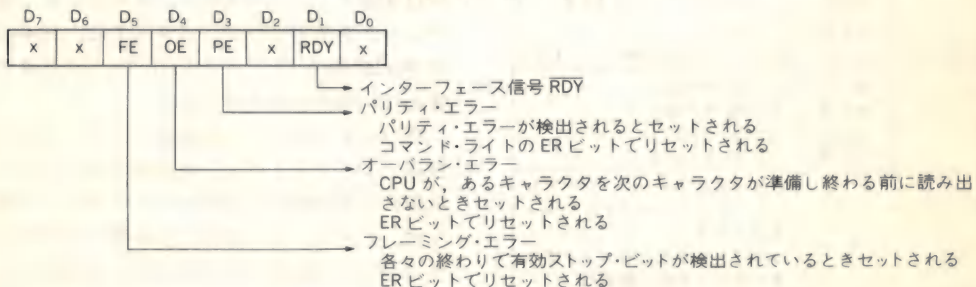
▶ D₄ : ER (ECL)

8251 のエラー・ステータス (PE, OE, FE) のクリア

〈図 2-49〉 コマンド・ライト



〈図 2-50〉 ステータス・リード



を行います。ER=1 でエラーはクリアされます。

► D₅: KBDE (RTS)

キーボード送信の許可/禁止を指示します。KBDE=0 で送信許可です。8251 の汎用出力ポート (RTS) の制御用です。

► D₆: IR (SRES)

8251 をソフトウェア・リセットさせます。IR=1 でリセットし、モード設定待ち状態になります。

► D₇: EH

8251 の同期モード用で、キーボード・インターフェースでは使用しません。

◆ ステータス読み出し (43H/リード)

図 2-50 はステータス・リードです。

► D₀: TxRDY

送信データ・バッファ状態を示します。TxRDY=0 でバッファにデータがあることを示し、この状態ではデータを送出できません。

► D₁: RxRDY

RxRDY=1 でデータを受信したことを示します。

キーボード・インターフェースでは、通常は割り込みで制御されますので、このポートを監視する必要はありません。

► D₂: TxEMP

送信データ・バッファ (第 2 バッファ) とトランスミ

ッタ内の送信バッファ (第 1 バッファ) が共に空であることを示します。

► D₃: PE

パリティ・エラーの発生を示します。エラーがあれば“1”，なければ“0”になります。エラーが発生しても 8251 の動作は停止しません。ER=1 でエラーはクリアされます。

► D₄: OE (OVE)

オーバラン・エラーの発生を示します。CPU が受信データの読み出しに遅れたときに“1”になります。エラーが発生しても 8251 の動作は停止しません。ER=1 でエラーはクリアされます。

► D₅: FE

フレーミング・エラーの発生を示します。ストップ・ビットが検出されなかったときに“1”になります。エラーが発生しても 8251 の動作は停止しません。ER=1 でエラーはクリアされます。

► D₆: SYNC/BRK

調歩同期モードでは、ブレイク信号 (RxD が 2 キャラクタ以上の時間“0”になった場合)を受信したときに“1”になります。キーボード・インターフェースでは使われません。

► D₆: DSR

汎用入力ポートの DSR の状態を示します。キーボード・インターフェースでは使われません。

〈図 2-51〉 PC98 シリーズのテキスト表示の機能概要

		ノーマル・モード	
表示文字種	ANK 文字, 特殊文字	244/246 字 (* 1)	
	JIS 第 1 水準漢字	2965 字 (* 2)	
	JIS 第 2 水準漢字	3384 字	
	非漢字	885 字	
	ユーザ定義文字	188(63) 字 (* 3)	
	拡張漢字	388 字	
	1/4 角文字 (* 4)	213 字	
表示文字容量	ANK	80 文字×25 行, 80 文字×20 行 40 文字×25 行, 40 文字×20 行	
	全角文字	40 文字×25 行, 40 文字×20 行	
表示文字構成	レターフェース	400ラインCRT	200ラインCRT
	全角	15×16	—
	半角	7×16	—
	ANK	7×13	6×8
	1/4 角	6×8	—
ドット数, 横×縦	ボディーフェース	400ラインCRT	200ラインCRT
	全角	16×16, 16×20	—
	半角	8×16, 8×20	—
	ANK	8×16, 8×20	8×8
アトリビュート	1/4 角	8×8, 8×10	—
	リバース, プリンク, シークレット, アンダ・ライン, パーチカル・ライン	カラー 8 色またはモノクロ濃淡キャラクタ単位に指定可	
	VRAM	テキスト表示用 4KB, 日本語表示用 4KB, アトリビュート 4KB 計 12KB CPU により直接 READ/WRITE GDC による描画機能なし パリティ・ビットなし	

- * 1: PC9801/E/F1, 2, 3/M2, 3/U2/VF2/VM0, 2, 4/UV2 には, バック・クォートとバック・スラッシュがない
 * 2: 日本語表示には, 専用高解像度ディスプレイ (400 ライン CRT) が必要
 * 3: PC9801 では, ユーザ定義文字は不可. PC9801E/F1, 2, 3/M2, 3/U2 では, ユーザ定義文字は 63 文字
 * 4: 1/4 角文字はグラフィック画面にのみ表示可能
 注: 98NOTE, PC9801BA, BX, PC9821, Ap, As, Ae, Ce, Af, PC9801P は専用高解像度ディスプレイ固定である

CRT ディスプレイ

◆ 機種による仕様の違い

PC98 シリーズには実に様々な機種があり, また機種によって CRT ディスプレイの機能に違いがあります。基本的には, 80 桁×25 行の漢字を表示できるテキスト表示と, 640×400×16 色のグラフィック表示を 2 枚ずつサポートしています。

PC9801/U…これらの機種ではグラフィック VRAM が 1 組しかありません。最近のプログラムでは, VRAM が 2 組あるものを前提に作られているものも多く, 異端的な機種になっています。

PC9801/E/F/M…これらの機種のグラフィック表

〈図 2-52〉 テキスト VRAM のメモリ空間

GDC アドレス	CPU アドレス ノーマル・モード	HIGH D ₁₅ ……D ₈	LOW D ₇ ……D ₀
0000	A0000	テキスト文字 1 ページ	テキスト文字 1 ページ
0800	A1000	テキスト文字 2 ページ	テキスト文字 2 ページ
1000	A2000	X	アトリビュート 1 ページ
1800	A3000		アトリビュート 2 ページ

↑ ワード・バイト・アドレス アドレス
 ↑ メモリは存在しない
 注: ハイレゾ・モードは省略

示は 8 色デジタル RGB のみ対応です。最近のグラフィックを多用したプログラムでは, 16 色 (4096 色中 16 色同時発色) アナログ RGB 対応が普通ですから, 使えない場合も多くあります。

PC98LT/HA…この機種はラップトップ・タイプで液晶ディスプレイによる画面表示を行います。また, 画面表示系は極めて特殊です。テキスト VRAM を持たず, グラフィック画面に字を書いて表示します。グラフィック画面も 1 枚 (2 色) しか出せず, 従来の PC98 シリーズとは大きく異なります。

その他のラップトップ, ノート系のコンピュータでは, 通常の PC98 シリーズと同じ仕様になっています。液晶ディスプレイによる制限で画面上では白黒 8 階調しか表示できないものもありますが (カラー表示可能な機種もある), その多くは, 外部 CRT ディスプレイ接続のための RGB インターフェースを持ちます。

最近の機種では 256 色モードを持つものがあります。PC9821/Ap/As/Ae/Ce/Af/Ne/Bp/Bs, PC98GS, PC386M 等では, 640×480×256 色を表示可能です (PC386M は 640×400×256 色)。

● 標準ディスプレイと専用高解像度ディスプレイ

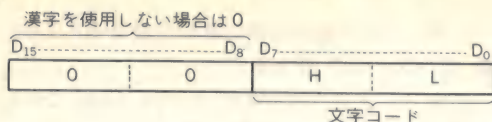
PC98 シリーズでは, 主に, 標準ディスプレイと専用高解像度ディスプレイで使用できます。標準ディスプレイは水平同期周波数が 15.75 kHz の 200 ライン用ディスプレイで, 400 ライン表示ができないために漢字表示がサポートされず, ほとんど使われません。

専用高解像度ディスプレイは水平同期周波数が 24.8 kHz の 400 ライン用ディスプレイで, 一般的に使用されているものです。また, 256 色 (480 ライン) モードが使用可能な機種では, 水平同期周波数が 31.5 kHz が使用可能なディスプレイを必要とします。

ハイレゾ・モードでは, 水平同期周波数が 32.8 kHz, または 50.0 kHz (インターレース) で, テキスト表示は 80 桁×25/31 行, グラフィック表示は 1120×750 を表示できます。

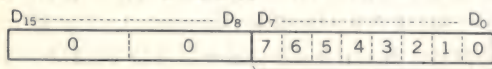
〈図 2-53〉 文字コード表現

● ANK



注: PC9801/E において漢字オプションを付けていない場合、D₈ から D₁₅ までには存在しない

● 簡易グラフ(ノーマル・モードのみ)

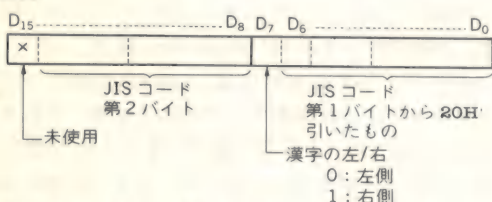


Mode F/Fbit0=0 で
アトリビュート・コードの
VL/G=1 のとき

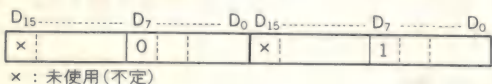
0	4
1	5
2	6
3	7

キャラクター
フェースの 8
分割に対応

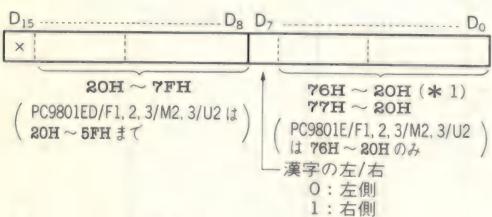
● 標準漢字



上記のように漢字の VRAM 上の表現は 4 バイトで行われる。
すなわち、次のような形式である



● ユーザ定義文字



* 1: 76H から 20H を引いた値

◆ テキスト表示

PC98 シリーズのテキスト表示の特色は、漢字の表示は英数字同様にテキスト VRAM への書き込みだけで表示されることです。PC/AT 互換機等の DOS/V のようにグラフィック VRAM に漢字パターンを書いて表示する機種に比べて、漢字を扱う表示が格段と速くなり、比較的速度の遅い CPU を載せた機種でも、全体として満足のいく処理速度が得られることが多いようです。

テキスト表示の機能概要を図 2-51 に示します。

● テキスト VRAM

テキスト VRAM は、CPU、GDC に対して図 2-52 のようなメモリ空間を持っています。

〈図 2-54〉 ANK 文字一覧

上位 4 ビット →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
下位 4 ビット ↓	0	D _E	0	@	P		p									
1	S _H	D ₁	!	1	A	Q	a	q								
2	S _X	D ₂	"	2	B	R	b	r								
3	E _X	D ₃	#	3	C	S	c	s								
4	E _T	D ₄	\$	4	D	T	d	t								
5	E _Q	N _K	%	5	E	U	e	u								
6	A _K	S _N	&	6	F	V	f	v								
7	B _L	E _B	'	7	G	W	g	w								
8	B _S	C _N	(8	H	X	h	x								
9	H _T	E _M)	9	I	Y	i	y								
A	L _F	S _B	*	:	J	Z	j	z								
B	H _M	E _C	+	:	K	[k	[
C	C _L	→	,	<	L	¥	l	l								
D	C _R	←	=	M]	m]]								
E	S _O	↑	>	N	^	n	^	^								
F	S _I	↓	/	?	O	_	o	_								

PC9801/E/F1, 2, 3/M2, 3/U2/VF2/VMO, 2, 4/U2 では 60H と FCH の文字はない

画面に表示される文字とテキスト VRAM の書くデータの関係は、**半角 1 文字に対して 2 バイト**(1 ワード/16 ビット)分を使用します。**D₀~D₇までの下位 4 バイトが文字コードを表し、D₈~D₁₅までの上位 4 バイトが文字の属性、漢字コードを表します。**

40 桁表示の場合は、テキスト VRAM のデータが**一つおきに有効**になります。例えば、1 桁 1 行目は A0000H~A0001H の 2 バイトで、2 桁 1 行目は A0004H~A0005H の 2 バイトになります。80 桁表示の場合は順番にすべての VRAM が有効です。

テキスト VRAM はすべて半角単位で指示します。**漢字表示の場合は半角 2 文字分(倍の大きさ)で漢字 1 文字を表現しますから、右部分、左部分を別々に指定して合計 4 バイト(2 ワード)必要になります。**

漢字表示のための漢字コードは、JIS コードが基本です。具体的には、下位 4 バイトには JIS コードの第 1 バイトから 20H を引いたものを、上位 4 バイトには JIS コードの第 2 バイトを書きます。また、漢字の左右どちらの部分を表示するかは、下位 4 バイト D₇ のビットで決まり、**D₇=0 で右側、D₇=1 で左側**を表示します。ユーザ定義が可能なユーザ定義文字も漢字と同様の方式で使用します。図 2-53 に文字コード表現を、図 2-54 に ANK 文字表示一覧を示します。

● アトリビュート

テキスト VRAM のすぐ上のアドレスにアトリビュート用の RAM があり、表示される文字の色や属性を決定します。アトリビュートはテキスト VRAM と 1 対 1 で対応し、同時に変更する必要があります。

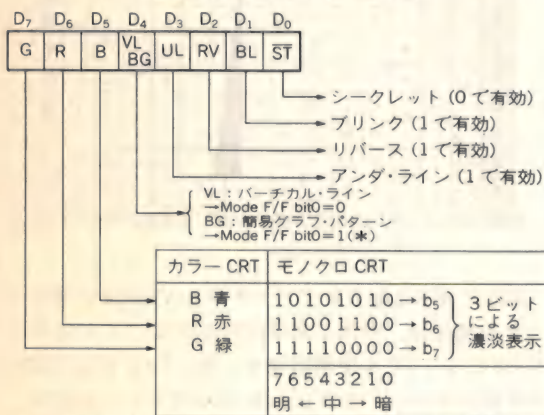
図 2-55 にアトリビュート表現、図 2-56 にアトリビュート表示を示します。

● カーソル表示

カーソルは基本的にはプリンキング・ブロック形式ですが、GDCの設定で自由な形式に設定することができます。

漢字表示のときのカーソル表示でカーソルが漢字の左側(1バイト目)にある場合は、漢字全体にカーソル表示されますが、カーソルが漢字の右側(2バイト目)にある場合は、漢字の右半分だけにカーソル表示されます。

〈図 2-55〉アトリビュート表現



※: 簡易グラフを出すときは b₄~b₇ をすべて 0 にする必要がある
 簡易グラフはノーマル・モードのみ

■ グラフィック表示

グラフィック表示の機能概要を図 2-57 に示します。

● グラフィック VRAM

グラフィック VRAM は、CPU、GDC に対して図 2-58 のようなメモリ空間を持っていて、モードや解像度に応じて、メモリの表示画面に対する割り付け方が異なります。VRAM は 32K バイトのプレーンが 4 枚 1 組 (16 色未対応機種では 3 枚) で、2⁴ で最大 16 色を同時に表示できます。また、同じ VRAM を 2 組持っていて、切り替えることで、同じアドレスから、表/裏 VRAM として使用できます (PC9801/U では VRAM は 1 組しかない)。

グラフィック VRAM は、図 2-59 に示すように、CPU からアクセスした場合と、GDC 経由でアクセスした場合は、データ・バス (メモリ) のビットの並びと、画面のドットの並びが逆になります。

グラフィックの表示には、640×200/400、カラー/モノクロといった画面モードがあります。これを変更するためには、GDC 等を図 2-60 のような設定にします。モノクロ・モードのグラフィック画面の合成では、パレットのレジスタの値を変更することで可能になります。

画面モードとハードウェアの関係を図 2-60、図 2-61 に示します。

〈図 2-56〉アトリビュート表示

ビット位置	名称	機能	ビット位置	名称	機能
0	シークレット ST	文字を表示しない。 ・ UL, VL は影響を受けない。 ・ 反転時はヌキ文字が消える。	3	アンダ ライン UL	・ ハイレゾ UL は必ず半カラム右にずれる。ただし、80 カラム目の右半分はカットされる。色指定により、その色が出る。
1	ブリンク BL	点滅表示を行う。 ・ UL, VL は点滅しない。 ・ 反転時はヌキ文字が点滅する。	4	バーチカル ライン VL	縦線を表示する。 ・ ノーマル ・ ハイレゾ 25 行 モード 20 行 モード
2	リバース RV	反転表示を行う。 ・ UL, VL は反転しない。	5	簡易グラフ パターン BG	簡易グラフ・パターンを表示する (ノーマル・モードのみ)
3	アンダ ライン UL	横下線を表示する。 ・ ノーマル	6	ブルー B	カラー CRT の色指定 青 2 ⁰
		UL は必ず半カラム右にずれる。ただし、80 カラム目の右半分はカットされる。色指定により、その色が出る。	7	レッド R	赤 2 ¹
				グリーン G	緑 2 ²

〈図 2-57〉 グラフィック表示の機能概要

ノーマル・モード	
モノクロ	640×200 ドット 16[12]画面×4画面×4組*合成可能
	640×400 ドット 8[6]画面×4画面×2組*合成可能
カラー	640×200 ドット 4画面*640×400 ドット 2画面*
	アナログ RGB ディスプレイ 使用時 4096 色中 16 色 (8 色) 表示 (16 階調濃淡表示可) デジタル RGB ディスプレイ 使用時 8 色中 8 色表示
VRAM	(32KB×4[3]プレーン)×2組*
	GDC による描画機能あり CPU による直接 READ/WRITE 可能 (GDC 描画中のアクセスは不可) グラフィック・チャージャ (GRGC, EGC) による READ/WRITE 可能 (チャージャ動作時, GDC の描画は不可)

注: PC98LT は VRAM が 32KB であり, 640×400 モノクロ・グラフィックのみ

4096 色中 16 色表示は, ディップ・スイッチ SW₁₋₈ ON 時 (拡張グラフィック・モード) のみ可能

* PC9801 および PC9801U2 では, 表示可能な画面数が半分になる

[] 内は 16 色表示未対応の場合の数値

PC9801/E/F1, 2, 3/M2, 3 では 16 色表示不可

PC9801U2/VF2/VM0, 2, 4 では 16 色グラフィック・ボードはオプション

PC9801/E/F1, 2, 3/M2, 3 にはグラフィック・チャージャ機能はない

PC9801BA, BX, P, 98NOTE, PC9821, Ap, As, Ae, Ce, Af は専用高解像度ディスプレイ固定である

〈図 2-58〉 グラフィック VRAM のメモリ空間

GDC アドレス	CPU アドレス	DATA HIGH DATA LOW				プレーン名			
		D ₁₅	D ₈	D ₇	D ₀	200 本表示		400 本表示	
						モノクロ・モード	カラー・モード	モノクロ・モード	カラー・モード
4000	A : 8000	P ₀₀ /P ₀₁ (※ 1) (GVRAM ₀)				PA ₀₀ /PA ₀₁	PA ₀ /PA ₁	PA ₀₀ /PA ₀₁	PA ₀ /PA ₁
						PB ₀₀ /PB ₀₁	PB ₀ /PB ₁		
8000	B : 0000	P ₁₀ /P ₁₁ (GVRAM ₁)				PA ₁₀ /PA ₁₁	PA ₀ /PA ₁	PA ₁₀ /PA ₁₁	
						PB ₁₀ /PB ₁₁	PB ₀ /PB ₁		
C000	B : 8000	P ₂₀ /P ₂₁ (GVRAM ₂)				PA ₂₀ /PA ₂₁	PA ₀ /PA ₁	PA ₂₀ /PA ₂₁	
						PB ₂₀ /PB ₂₁	PB ₀ /PB ₁		
(* 2) 00000	E : 0000	P ₃₀ /P ₃₁ (GVRAM ₃)				PA ₃₀ /PA ₃₁	PA ₀ /PA ₁	PA ₃₀ /PA ₃₁	PA ₀ /PA ₁
						PB ₃₀ /PB ₃₁	PB ₀ /PB ₁		

→ バイト・アドレス (アドレスの最下位により H, L バイトを切り分ける)

→ ワード・アドレス (H, L バイトは同時にアクセス)

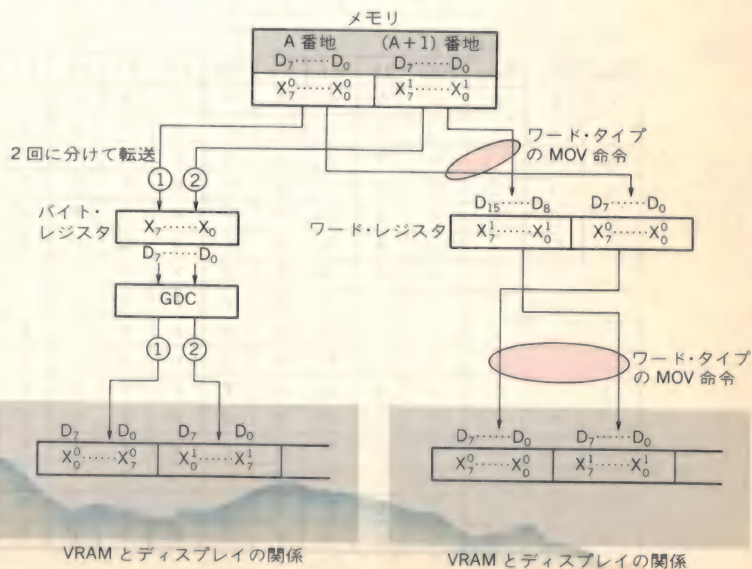
* 1: 上記空間において, PC9801/U2 では P₀₁, P₁₁, P₂₁, P₃₁ は存在しない

* 2: 8 色モードのときはバッファが Enable にならない

I/O ポート (42H) bit3=1 のときは 16 色表示対応

〈図 2-59〉

グラフィック VRAM へのアクセスの方法



VRAM とディスプレイの関係

VRAM とディスプレイの関係

〈図 2-60〉 画面モードとハードウェアの関係

表 示 状 態				設 定 値						
CRT	グラフィック ・モード	グラフ 解像度	表 示 ブ レ ー ン (* 2)	GDC L/F	GDC L/R	GDC SAD	Palette Reg	Mode F/F bit 1	Mode F/F bit 4	
専用高解像度 ディスプレイ	カラー	640× 200	PA _i	400	2	0	各コード の RGB	0	(* 1) 1	
			PB _i			1F40H				0
		640× 400	PA _i		1	0				
	モノクロ	640 ×	PA _{0i} PA _{1i} PA _{2i} (PA _{3i})		2	0	画面 合成 コード	1	(* 1) 1	
			PB _{0i} PB _{1i} PB _{2i} (PB _{3i})							1F40H
		200			1	0			0	
		640 × 400	PA _{0i} PA _{1i} PA _{2i} (PA _{3i})							
	高解像度 および 標準 ディスプレイ	カラー	640× 200	PA _i	200	1	0	各コード の RGB	0	0
			PB _i	1F40H						
モノクロ		640 × 200	PA _{0i} PA _{1i} PA _{2i} (PA _{3i})	0			画面 合成 コード	1		
			PB _{0i} PB _{1i} PB _{2i} (PB _{3i})						1F40H	

()内は、16色表示未対応時は無効

*1: 1を設定した場合は画面が1本おきの表示になる
0を設定した場合は画面に2本同じ走査線が表示される

*2: PA_i, PA_iのi=1のとき、I/Oポート・アドレス 0A4Hに 01HをOUTする
i=0のとき、I/Oポート・アドレス 0A4Hに 00HをOUTする
PC9801/U2の場合、表示ブレーンはi=0ブレーンのみ使用可能

*3: PC9801BA, BX, P, 98NOTE, PC9821, Ap, As, Ae, Ce, Afは専用高解像度ディスプレイ固定である

8色モード

表示ブレーン			パレット・レジスタの値							
PA _{0i}	PA _{1i}	PA _{2i}	#0	#1	#2	#3	#4	#5	#6	#7
×	×	×	0	0	0	0	0	0	0	0
×	×	○	0	0	0	0	7	7	7	7
×	○	×	0	0	7	7	0	0	7	7
○	×	×	0	7	0	7	0	7	0	7
×	○	○	0	0	7	7	7	7	7	7
○	×	○	0	7	0	7	7	7	7	7
○	○	×	0	7	7	7	0	7	7	7
○	○	○	0	7	7	7	7	7	7	7

〈図 2-61〉

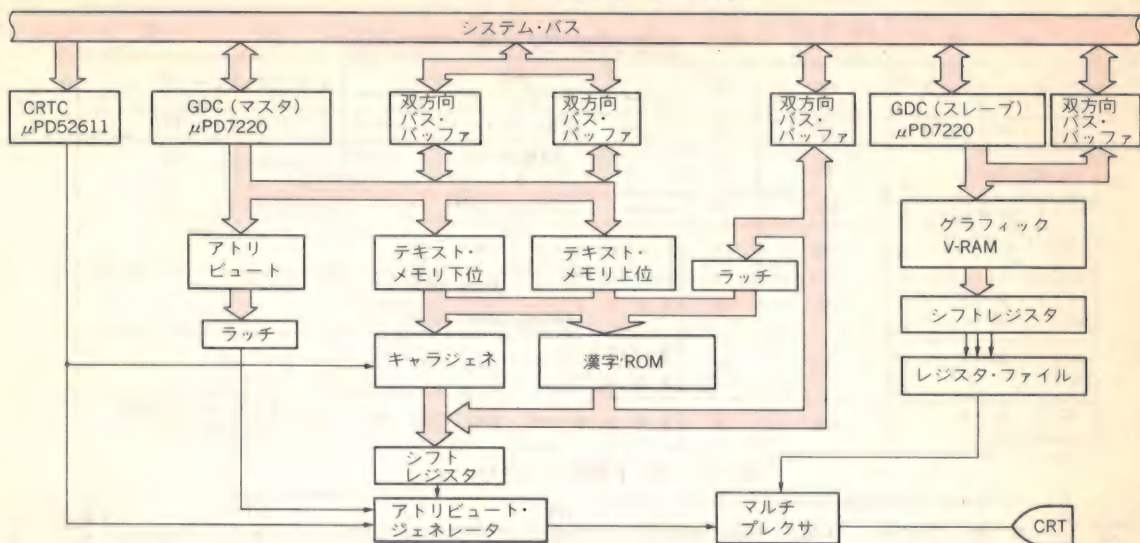
画面合成コード(モノクロ・モード)

* ×印は画面合成に関係しないブレーン

16色モード

表示ブレーン				パレット・レジスタの値															
PA _{0i}	PA _{1i}	PA _{2i}	PA _{3i}	#0	#1	#2	#3	#4	#5	#6	#7	#8	#9	#A	#B	#C	#D	#E	#F
×	×	×	×	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
×	×	×	○	0	0	0	0	0	0	0	0	F	F	F	F	F	F	F	F
×	×	○	×	0	0	0	0	F	F	F	F	0	0	0	0	F	F	F	F
×	×	○	○	0	0	0	0	F	F	F	F	F	F	F	F	F	F	F	F
×	○	×	×	0	0	F	F	0	0	F	F	0	0	F	F	0	0	F	F
×	○	×	○	0	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F
×	○	○	○	0	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F
○	×	×	×	0	F	0	F	0	F	0	F	0	F	0	F	0	F	0	F
○	×	×	○	0	F	0	F	0	F	0	F	0	F	0	F	0	F	0	F
○	×	○	×	0	F	0	F	F	F	F	F	0	F	F	F	F	F	F	F
○	×	○	○	0	F	0	F	F	F	F	F	F	F	F	F	F	F	F	F
○	○	×	×	0	F	F	F	0	F	F	F	F	F	F	F	F	F	F	F
○	○	×	○	0	F	F	F	0	F	F	F	F	F	F	F	F	F	F	F
○	○	○	×	0	F	F	F	F	F	F	F	0	F	F	F	F	F	F	F
○	○	○	○	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F

<図 2-62> CRT コントロール・ブロック



GDC と周辺 LSI

►使用 LSI

 μ PD7220 相当 $\times 2$, μ PD52611

▶ I/O アドレス

60H, 62H (GDC/マスタ)

AOH, A2H (GDC/スレーブ)

70H, 72H, 74H, 76H, 78H, 7AH (CRTC)

◆ GDC と周辺 LSI のハードウェア

PC98 シリーズでは、CRT インターフェースに GDC(μ PD7220)が2個使用されています。そのうち1個はテキスト画面用でマスタ動作で、もう一つはグラフィック画面用でスレーブ動作になっています。マスタの GDC は、60H、62H、スレーブの GDC は A0H、A2H で操作できます。他にもタイミング関係用に CRT C(μ PD52611)等が使用されています。

垂直同期信号を出しているマスタの GDC は、テキスト VRAM のアドレス生成、CRT の同期信号の生成を行い、CRT 関係回路の核になっています。スレーブの GDC はマスタからの同期信号に合わせて、グラフィック画面を制御します。

VRAMからの出力信号は、テキストならばキャラクタ・ジェネレータ(CG・漢字ROM等)を通り、グラフィックならばそのままシフトレジスタでパラレル-シリアル変換され、必要に応じてパレット操作や、アナログRGBのためのD-Aコンバータを経てビデオ信号になります。

CRTC は、主にテキスト用の垂直同期関係の制御を行い、CG ライン・カウンタの出力、アドレス加算

回路へのタイミング出力、アングライン等のタイミング出力を行っています。それらの制御用に「ライン・カウンタ制御命令」として、70H~7AH までの6個のレジスタを持ちます。

CRT コントロールのブロック図を図2-62に示します。

◆ テキスト表示制御用命令

テキスト表示関連の I/O アドレスとしては以下の
ようなものがあります。

60H, 62H	GDC(マスタ)
64H	CRT インタラプト・リセット
68H	モード・レジスタ1 設定
6CH	ボーダ・カラー設定
6AH	モード・レジスタ2 設定

CRT インタラプト・リセット 64H は、スムーズ・スクロール制御のための割り込みリセットです。CRTV (VSYNC) 割り込みを発生するためのもので、64H にライト(リセット)すると、一度だけ CRT の VSYNC 信号の立ち上がり時に「ベクタ番号 0AH」の割り込みがかかります。通常は連続して割り込みを使用するので、CRTV 割り込みプログラムの中で、このポートをアクセス(リセット)します。この場合、約 16 ms 間隔で割り込みが発生します。

CRTV 割り込みは、電源投入後に1回だけ割り込むだけで、リセットしない限り、その後は割り込みません。

モード・レジスタ 1 設定 68H は、主に画面表示に関連するモードの変更をします。設定は ADR₀₋₂ に変更したい項目、DT には変更したいモードを入れて書き込みます。

▶ ATR SEL

〈図 2-63〉 テキスト表示制御命令

命 令	I/O ポート・アドレス	R/W	デ ー タ								備 考
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
リード・ステータス	60	R	← GDC ステータス・フラグ →								μPD7220(スレーブ)
ライト・パラメータ	60	W	← GDCパラメータ →								μPD7220(スレーブ)
リード・データ	62	R	← GDCデータ (ライト・ペン) →								μPD7220(スレーブ)
ライト・コマンド	62	W	← GDCコマンド →								μPD7220(スレーブ)
CRT インタラプト・リセット	64	W	×	×	×	×	×	×	×	×	
ライト・モード・レジスタ(1)	68	W	0	0	0	0	← Mode F/F → ADR ₂ ADR ₁ ADR ₀ DT				
ライト・ボーダ・カラー	6C	W	← ボーダ・カラー → 0 G R B		0	0	0	0			
ライト・モード・レジスタ(2)	6A	W	0	0	0	0	0	EX ₁	EX ₂	DT	16色/8色モード切り替え EGC/GRCG 切り替え

〈図 2-64〉 モード設定レジスタ(モード 1)

Mode F/F	名 前	ADR ₂	ADM ₁	ADR ₀	DT		主として関係する部分
					1	0	
0	ATR SEL	0	0	0	ATR7 が簡易グラフ	ATR7 がパーティカル・ライン	テキスト
1	GRAPHIC Mode	0	0	1	モノクロ・グラフィック・モード	カラー・グラフィック・モード	グラフ
2	Column WIDTH	0	1	0	40 字モード	80 字モード	テキスト
3	FONT SEL	0	1	1	文字フォントの大きさ		テキスト
					7×13	6×8	
4	GRP Mode	1	0	0	専用高解像度ディスプレイを 200 本モード・グラフで使用する	・専用高解像度 400 本 ・標準解像度	グラフ
5	KAC Mode	1	0	1	漢字アクセス・モード		漢 字
					ビット・マップ	コード・アクセス	
6	NVMW PERMIT	1	1	0	不揮発メモリへの書き込み		
					PERMIT	INHIBIT	
7	DISP ENABLE	1	1	1	表示可とする	すべての画面を表示しない	テキスト

テキスト VRAM のアトリビュートの D₄ で設定する簡易グラフと、パーティカル・ラインの切り替えをします。

▶ GRAPHIC Mode

モノクロとカラーの切り替えをします。

▶ Column WIDTH

40 桁/80 桁の切り替え、40 桁時は一つおき表示されます。

▶ FONT SEL

表示されるフォントを切り替えます。6×8 モードでは漢字の表示ができません。

▶ GRP Mode

専用高解像度ディスプレイで 200 ライン表示モード

を使用するときに「1」にします。400 ライン・モード時に使用すると、1 行おきに表示されます。

▶ KAC Mode

漢字アクセス・モードを変更します。CG ウィンドウやキャラクタ・ジェネレータ制御命令で使します。

▶ NVMW PERMIT

不揮発メモリへの書き込みを許可/禁止を指定します(メモリ・スイッチ)。

▶ DISP ENABLE

画面の表示の許可/禁止を指定します。

「ボーダ・カラー設定(6CH)」は、描画面の外側のカラーを変更します。専用高解像度ディスプレイでは、描画面の上下の色が出ません。

「モード・レジスタ 2 設定(6AH)」は、16 色/8 色モードの切り替えをします。

EGC の拡張モード指定にも使用されます。

テキスト表示制御命令を図 2-63 に、モード設定レジスタ 1 を図 2-64 に、モード設定レジスタ 2 を図 2-65 に示します。

〈図 2-65〉 モード設定レジスタ(モード 2)

ADR								名 前	DT	
6	5	4	3	2	1	0			1	0
0	0	0	0	0	0	0		COLOR SEL	16 色モード	8 色モード

ハードウェア・リセット時 0 側(8 色モード)になる

〈図 2-66〉 ライン・カウンタ制御命令

命 令	I/O ポート ・ アドレス	R/W	データ	備 考
			D ₇ D ₆ D ₅ D ₄ D ₃ D ₂ D ₁ D ₀	
ライト PL	70	W	キャラクタ位置ライン 数 (PL) (*1)	ライン・カウンタの初期値 (ボディーフェースのうち、キャラクタが表示される位置の上から数えたライン数の 2 の補数)
ライト BL	72	W	ボディーフェース・ライン 数 (BL) (*1)	キャラクタの先頭を 0 としたときのボディーフェース下端のライン数
ライト CL	74	W	キャラクタ・ライン数 (CL) (*1)	キャラクタ・フェースのライン数
ライト SSL	76	W	スムーズ・スクロール・ライン数 (SSL)	スクロール・エリア内の文字がスクロールしているライン数
ライト SUR	78	W	スクロール・エリア上 辺位置行数 (SUR)	スクロール・エリアの上辺の位置の行数の 2 の補数 (この次の行よりスクロールする)
ライト SDR	7A	W	スクロール・エリア行 数 (SDR)	(スクロール・エリアの行数) - 1

		25 行	20 行
PL	専用高解像度 ディスプレイ	00H	1EH
	高解像度および 標準ディスプレイ	00H	1FH
BL	専用高解像度 ディスプレイ	0FH	11H
	高解像度および 標準ディスプレイ	07H	08H
CL	専用高解像度 ディスプレイ	10H	
	高解像度および 標準ディスプレイ	08H	

* 1 : 初期設定値

◆ ライン・カウンタ制御用命令

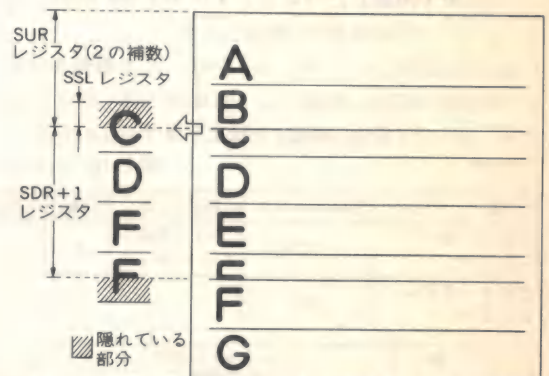
ライン・カウンタ制御回路は CG のライン・カウンタ出力やアンダラインのタイミング出力、スムーズ・スクロール機能を実現するためのアドレス加算回路へタイミング出力等、CRT の垂直方向の制御信号を出力します(図 2-66)。

◆ スムース・スクロール制御用命令

スムーズ・スクロールは主にライン・カウンタ制御回路によって実現されています。スムーズ・スクロールさせるためには、**スクロール・エリアの上辺行数 (SUR)**、**スクロールする行数 (SDR)** をセットして、画面描画のブランキングのタイミング (CTRV/SYNC 割り込み) で **スムーズ・スクロール・ライン数 (SSL)** を増減させることで、ちらつきがなく、画面がスムーズにアップ・ダウンできます。

スムーズ・スクロール・エリア (SUR, SDR で設定) では、SSL だけ上にずれて (加算されて) 表示されます。スクロール・エリアの下辺行では、その 1 行下の文字の上の部分が表示されてしまいますので、下辺行とその下の行の文字がだぶって表示されることになります。これを防ぐためには、**スクロール・エリアより下の部分を、GDC の SCROLL コマンド等で画面を分割して、スクロール・エリアと分離させておきます**。図 2-67 がスムーズ・スクロールのようすです。

〈図 2-67〉 スムース・スクロールのようす



※画面最上位行からスムーズ・スクロールさせる場合は
SUR=1FH, 2 行目からなら SUR=1EH になる

AAH パレット・レジスタ (GREEN)
ACH パレット・レジスタ (RED)
AEH パレット・レジスタ (BLUE)

表示画面選択レジスタ (A4H) は、2 画面 (裏/表) ある VRAM のうち、表示する画面を指定します。00H で表 VRAM (P₀₀, P₁₀, P₂₀, P₃₀)、01H で裏 VRAM (P₀₁, P₁₁, P₂₁, P₃₁) を選択します。

描画画面選択レジスタ (A6H) は、2 画面 (裏/表) ある VRAM のうち、描画 (CPU の読み書き) する画面を指定します。00H で表 VRAM (P₀₀, P₁₀, P₂₀, P₃₀)、01H で裏 VRAM (P₀₁, P₁₁, P₂₁, P₃₁) を選択します。表示画面と描画画面を別々に設定できるので、描画中に、表示画面のちらつきをなくせます。

パレット・レジスタ (A8H~AEH) は、8 色モードと 16 色モードで設定方法が違います。

▶ 8 色モード時

VRAM で指定したパレット・コード (8 色) を、別のカラー・コード (8 色) へ変換します。

一つのパレット・コードにつき、一つのパレット・レジスタを持ち、カラー・コードを 3 ビットで表します。

◆ グラフィック表示制御用命令

テキスト表示関連の I/O アドレスとしては以下のようなのがあります。

A0H, A2H GDC (スレーブ)
A4H 表示画面選択レジスタ
A6H 描画画面選択レジスタ
A8H パレット・レジスタ (パレット番号)

〈図 2-68〉 パレット・レジスタ 8 色モード

パレット・コード	アドレス	カラー・コード (R=赤, G=緑, B=青)							
		D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
0	AEH	0	B	G	R	×	×	×	×
1	AAH	0	B	G	R	×	×	×	×
2	ACH	0	B	G	R	×	×	×	×
3	A8H	0	B	G	R	×	×	×	×
4	AEH	×	×	×	×	0	B	G	R
5	AAH	×	×	×	×	0	B	G	R
6	ACH	×	×	×	×	0	B	G	R
7	A8H	×	×	×	×	0	B	G	R

A8H～AEH までの 4 バイトのレジスタを、上位 4 ビット、下位 4 ビットに分けて使用して、合計 8 個分のパレット・レジスタとして使います。4 ビット中上位 1 ビットは未使用です (図 2-68)。

▶ 16 色モード時

VRAM で指定したパレット・コード (16 色) を、カラー・コード (4096 色) へ変換します。

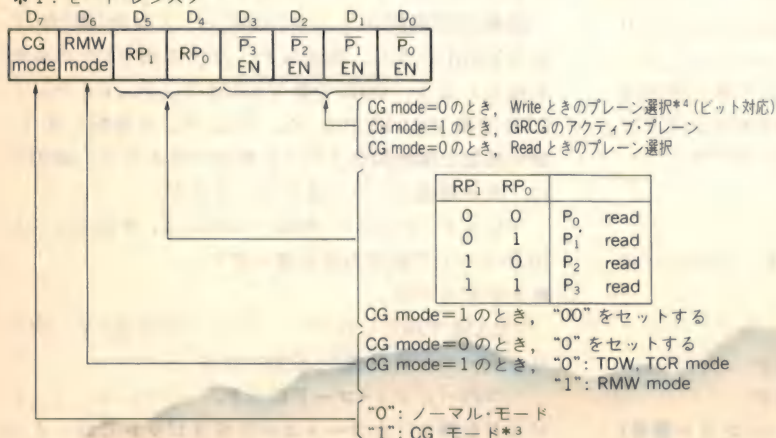
A8H に変更したいパレット・コードの番号を入れて、AAH, ACH, AEH に、RGB (赤・緑・青) 別にカラー・コード (各色の輝度) を設定します (図 2-69)。

〈図 2-70〉 グラフィック・チャージャ制御命令

命 令	I/O アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
ライト・モード・レジスタ	7C	CG モード	RMW モード	0	0	P ₃ EN	P ₂ EN	P ₁ EN	P ₀ EN
ライト・タイル・レジスタ	7E	タイル・レジスタ 0～3 (*)							

ビット名	ビット=1 の意味	ビット=0 の意味
CG モード	GRCG を有効とする CPU の VRAM アクセスをきっかけとして、GRCG が各モードの動作を実行する	GRCG を無効とする CPU の VRAM アクセスは、そのまま VRAM のリード (ライト) となる
RMW モード	CPU の VRAM ライトにより、RMW モードの動作を行う CPU の VRAM リードは無視される	CPU の VRAM ライトにより TDW モードの動作を行う CPU の VRAM リードにより TCR モードの動作を行う
P ₃ EN, P ₂ EN P ₁ EN, P ₀ EN	該当するプレーンを無効とする	該当するプレーンを有効とする 複数ビットの指定が可能 GRCG は、有効となっているプレーンに対してのみアクセスを行う

* 1: モード・レジスタ



〈図 2-69〉 パレット・レジスタ 16 色モード

アドレス	内 容	データ
A8H	パレット番号	00H-0FH
AAH	GREEN の輝度	00H-0FH
ACH	RED の輝度	00H-0FH
AEH	BLUE の輝度	00H-0FH

◆ グラフィック・チャージャ (GRCG) 制御命令

GRCG は、CPU と VRAM の間に専用のハードウェアを入れることで、一度の読み書きで、4 枚ある VRAM すべてにアクセスできるので、グラフィック描画を高速に行えます。特に画面を同じ色で塗る場合には効果を発揮します。GRCG は、PC9801/E/F/M では使用できません。

GRCG の制御レジスタは二つあり、これで VRAM アクセス時のモード (効果) を変更します。

GRCG 制御命令を図 2-70 に示します。

▶ TDW モード

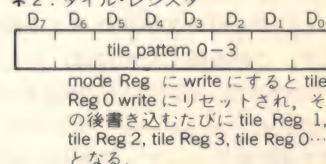
CPU が VRAM に書き込むと CPU のデータは無視され、タイル・レジスタの内容 (8 ビット・レジスタが

*: モード・レジスタにライトを行うと、タイル・レジスタ 0 ライトにリセットされる。その後ライトするたびに、書かれるレジスタはレジスタ 1, レジスタ 2, レジスタ 3, レジスタ 0, と変化する

* GRCG 動作中は、CPU に WAIT がかかり、モード・レジスタの変更が不可となる

* GRCG 動作中は、バスが占有されるため DMA 転送レートが低下する。原則として DMA と GRCG は同時に使用しないこと (例えば、固定ディスクの DMA 転送などと、GRCG に対するストリング命令によるアクセスは同時に行わないようにする必要がある)

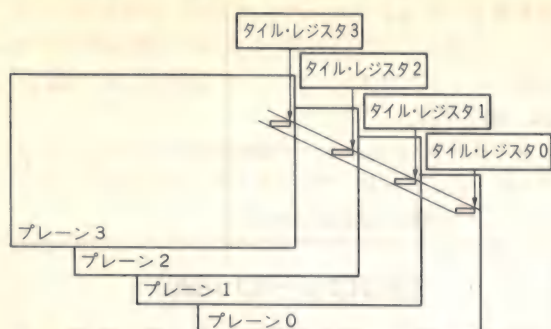
* 2: タイル・レジスタ



* 3: グラフィック・チャージャ動作中 (CG mode) は GDC からの描画は禁止される。GDC に描画命令を出しても、VRAM への直接アクセスは起こらない。モード・レジスタはグラフィック・チャージャ動作中は変更できない

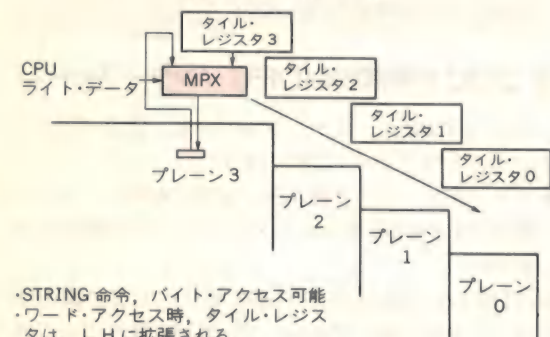
* 4: この設定は CG mode=0 のときは GDC の描画に対しても有効 (READ プレーンと WRITE プレーンを変えて設定すると、プレーン間の転送が可)

〈図 2-71〉 TDW モード



- ・STRING 命令, バイト・アクセス可能
- ・ワード・アクセス時, タイル・レジスタ・パターンはL,Hに拡張される。

〈図 2-73〉 RMW モード



- ・STRING 命令, バイト・アクセス可能
- ・ワード・アクセス時, タイル・レジスタは, L, Hに拡張される。

0	1	1	0	0	1	1	0	アクセス前のデータ
1	0	1	0	1	0	1	0	タイル・レジスタの値
0	0	0	0	1	1	1	1	CPU ライト・データ
0	1	1	0	1	0	1	0	マアクセス後の VRAM データ

4 個)を VRAM に書き込みます。画面を同じ色で塗る場合には効果があります(図 2-71)。

▶ TCR モード

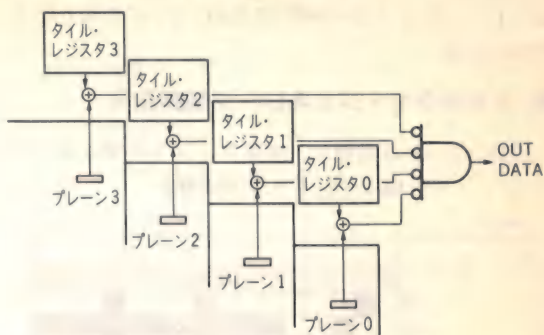
CPU が VRAM を読み出すと、各プレーンとタイル・レジスタの内容を比較し、一致したビットを「1」として CPU に読み込みます。画面の色を識別するのに効果があります(図 2-72)。

▶ RMW モード

CPU が VRAM に書き込むと、CPU のデータのうち、ビットが 1 である部分は、タイル・レジスタの内容が VRAM に書かれ、ビットが 0 である部分は、元のデータのまになります。TDW モードをビット単位で行うことができますので、ドットやラインを引く場合には有効です(図 2-73)。

リスト 2-5 に GRGCG テストの参考プログラムを示します。

〈図 2-72〉 TCR モード



- ・SCAN 命令, バイト・アクセス可能。
- ・ワード・アクセス時, タイル・レジスタ・パターンは, H, Lに拡張される。
- ・色の検出ではタイル・レジスタをすべて 1 かすべて 0 にセットする。
- ・アクティブにしないプレーンは無視される。

〈リスト 2-5〉 グラフィック・チャージャのテスト・プログラム

```

/*
** GRGCGテスト
** VRAM消去
**
*/
#include <dos.h>

int main()
{
    int i;
    unsigned int adres;

    outportb(0x7c, 0x80); /* TDWモード、全プレーン有効 */
    for (i = 0; i < 4; i++) { /* タイルレジスタ設定 */
        outportb(0x7e, 0);
    }

    for (adres = 0; adres < 32*1024; adres+=2) {
        poke(0xa800, adres, 0);
    }
    outportb(0x7c, 0); /* GRGCG無効 */

    return 0;
}

```

〈図 2-74〉 CG ウィンドウ・アドレス

命 令	I/O アドレス	R/W
Write 2nd byte code	0A1H	W
Write 1st byte code	0A3H	W

■ CG ウィンドウ

CG ウィンドウは、メモリ空間の一部(A4000H ~A4FFFFH)に、文字フォントのデータを出して、読み書きができます。フォント・データが CG ウィンドウのメモリ空間より大きいために、フォント・データの一部分をウィンドウ(窓)から見る形で使います。

PC9801VM 以前の機種や、ラップトップ、ノート等では、CG ウィンドウを搭載していない機種もあります。CG ウィンドウ・アドレスを図 2-74 に、データの読み書きを図 2-75 に示します。

CG ウィンドウで漢字フォントを読み書きする場合には、ビット・マップ・モードとコード・アクセス・モードがあります。コード・アクセス・モードは高速に読み書きできますが、GDC が V-SYNC 中でないと画

面が乱れることがあります。この切り替えはライト・モード・レジスタ(1)/68HでKACモードの切り替えで行います。

■ キャラクタ・ジェネレータ制御命令

CG ウィンドウ同様に、文字フォントの読み出しに
〈図 2-75〉データの読み書き



第1バイトが以下のものを除く
・2C~2F: 全角ケイ線特殊文字
・76: ユーザ定義
・79~7C: 拡張漢字

〈図 2-76〉キャラクタ・ジェネレータ制御命令

命 令	I/O ポート・ アドレス	R/W	データ							
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
ライト文字コード 第2バイト	A1	W	文字コード 第2バイト							
ライト文字コード 第1バイト	A3	W	文字コード 第1バイト							
ライト・ライン・ カウンタ(*1)	A5	W	0	0	L	R	R	R	R	R
					C	C	C	C	C	C
					R	4	3	2	1	0
リード文字パターン 〈図2-77参照〉	A9	R	CG リード・パターン ← 左 右 →							
ライト文字パターン 〈図2-77参照〉	A9	W	CG ライト・パターン ← 左 右 →							

* 1: L/R は全角漢字左(1), 右(0)を指定する。
RC₀~RC₄はキャラクタ・パターンの上から何ラインかを16進
数で指定する。ただし、CGのパターンは16ライン固定なので
RC₄は0としておく(未定義でよい)

〈図 2-78〉GDC と CPU のインターフェース

アドレス	モード	機 能
60H/AOH	リード	GDC ステータス・フラグの読み出し
60H/AOH	ライト	パラメータの書き込み
62H/A2H	リード	データの読み出し
62H/A2H	ライト	コマンドの書き込み

〈図 2-79〉
GDC ステータス・レジスタ

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
LIGHT PEN DETECT	HORIZONTAL SYNC	VERTICAL SYNC	DMA EXECUTIVE	DRAWING	FIFO EMPTY	FIFO FULL	DATA READY

使用します。CG ウィンドウがフォント全体を一度に
読み書きできるのに比べて、これは、一度に16ビット
分しか読み出せないの、1文字分の読み書きでも、
何度かライト・ライン・カウンタを操作します(図 2-
76, 図 2-77)。

キャラクタ・ジェネレータ制御命令にもビット・マップ
・モードとコード・アクセス・モードがあり、CG ウィ
ンドウと同様の制約があります。

GDC(μPD 7220)

▶ I/O アドレス

60H, 62H(GDC/マスタ)

AOH, A2H(GDC/スレーブ)

■ GDC の制御方法(CPU インターフェース)

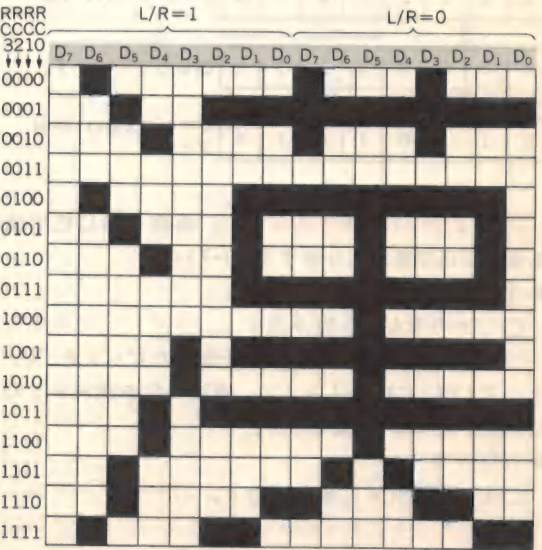
GDC と CPU のインターフェースは、図 2-78 に示
す二つの I/O アドレスで行います。

▶ ステータス・フラグ読み出し(60H/AOH・リード)

図 2-79 に GDC のステータス・レジスタの構成を示
します。

D₀: DATA READY……GDC がリード等の読み出
しコマンド実行後、読み出しデータが、読み出し可能
な状態になったことを示します。

〈図 2-77〉データの読み書き



〈図 2-80 (a)〉 GDC のコマンド/パラメータ (動作制御)

コマンド名	機 能	C/P	コマンドまたはパラメータ・コード								パラメータ解説
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
RESET	初期化	C	0	0	0	0	0	0	0	0	
SYNC	動作モード、同期タイミングの定義	C	0	0	0	0	1	1	1	DE	DE=0:表示停止、DE=1:表示開始
		P ₁	0	0	CHR	F	I	D	G	S	CHR=1:文字モード、F=1:フラッシュレス描画 I=1:インターレース・モード、D=1:ダイナミックRAM G=1:グラフィック・モード、S=1:インターレース・シュリンク・モード (I=1)
		P ₂	C/R								C/R: 1 行の表示文字数
		P ₃	VSL				HS				HS: 水平同期期間
		P ₄	HFP				VS _H				VS: 垂直同期期間
		P ₅	HBP								HFP: 水平右側非表示期間
		P ₆	VFP								HBP: 水平左側非表示期間
		P ₇	L/F _L								VFP: 垂直上側非表示期間
		P ₈	VBP								L/F _H
MASTER/SLAVE	マスタ動作、スレーブ動作の選択	C	0	1	1	0	1	1	1	M	M=1:マスタ動作、M=0:スレーブ動作

〈図 2-80 (b)〉 GDC のコマンド/パラメータ (映像メモリ制御)

コマンド名	機 能	C/P	コマンドまたはパラメータ・コード								パラメータ解説
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
WRITE	ドット修正モード設定 データの書き込み 繰り返し送出可 バイト転送時はCODE _L のみでよい	C	0	0	1	WLH	0	MOD		WLH=00:ワード転送 MOD=00:REPLACE =01:未定 =01:COMPLEMENT =10:下位バイト転送 =10:CLEAR =11:上位バイト転送 =11:SET	
		P ₁	CODE _L								CODE:V-RAM 書き込みデータ
		P ₂	CODE _H								
READ	映像メモリの読み出し	C	1	0	1	WLH	0	MOD		WRITE コマンドに同じ	
DMAW	映像メモリへのDMA転送指示	C	0	0	1	WLH	1	MOD		WRITE コマンドに同じ	
DMAR	映像メモリからのDMA転送指示	C	1	0	1	WLH	1	MOD		WRITE コマンドに同じ	

D₁: FIFO FULL……FIFO がデータでいっぱいになったことを示します。

D₂: FIFO EMPTY……FIFO が空であることを示します。

D₃: DRAWING……GDC が描画中であることを示します。

グラフィックの描画時には描画開始から終了まで“1”になっていますが、テキスト描画時には内蔵 RAM から GDC 内部のレジスタに、その内容が転送されるたびに“0”になります。また、DMAR/DMAR/READ/WRITE コマンド実行中は GDC が描画サイクルにあるときだけ“1”になります。

D₄: DMA EXECUTE……DMA 転送を続行中であることを示します。PC98 シリーズでは GDC と DMA が接続されていないために、DMA は使用できません。

D₅: VERTICAL SYNC……垂直同期信号が発生していることを示します。

D₆: HORIZONTAL BLANK……水平同期信号が発生していることを示します。

D₇: LIGHT PEN DETECT……ライト・ペン信号によるアドレスの検出があったことを示します。

▶パラメータ書き込み (60H/A0H・ライト) とコマン

ド書き込み (62H/A2H・ライト)

FIFO FULL=0であることを確認したうえで書き込みます。または、FIFO EMPTY=1であることを事前に確認をして、16 バイトまでのコマンドを一気に送ることもできます。17 バイト目以降は FIFO FULL を確認しながら送ります。

▶データ読み出し (62H/リード)

DATA READY=1であることを確認したうえで読み出します。FIFO は、RESET コマンド直後は CPU → GDC の方向になっています。しかし、READ/CSRR/LPEN 等のコマンドが実行されたとき GDC → CPU の方向へ変化します。したがって、これらのコマンドの直後は、CPU → GDC 方向の FIFO は動きません。この 3 種類以外のコマンドを実行すると、CPU → GDC の方向へ戻ります。

◆ GDC のコマンド (動作制御)

図 2-80 に GDC コマンド/パラメータ一覧を示します。

▶ RESET

初期化コマンドです。基本タイミング回路、FIFO のクリア、描画動作停止、画面表示停止等です。

〈図 2-80(c)〉 GDC のコマンド/パラメータ (表示制御)

コマンド名	機 能	C/P	コマンドまたはパラメータ・コード								パラメータ解説	
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
START	表示開始	C	0	1	1	0	1	0	1	1	} どちらでもよい	
STOP	表示停止	C	0	0	0	0	1	1	0	1		
ZOOM	拡大係数の設定	C	0	1	0	0	0	1	1	0	ZR: 拡大表示時の拡大係数 ZW: 拡大描画時の拡大係数	
		P	← ZR →				← ZW →					
SCROLL	表示開始アドレス、表示領域の設定 画面を n 個に分割した場合、このパラメータを n 回送出、 文字モード: $n_{\max}=4$ 文字/グラフィック・モードと $n_{\max}=2$ グラフィック・モード	C	0	1	1	1	← RA →				RA: データ RAM のアドレス	
		P _{n1}	← SAD _{nL} →				← SAD _{nH} →				SAD _n : n 番目の区間の開始アドレス	
		P _{n2}	0	0	0	← SAD _{nH} →				SL _n : n 番目の区間の長さ		
		P _{n3}	← SL _{nL} →				0	0	0	0	IM: 文字/グラフィック・モード時 IM = 0... 文字表示領域 (その他のモード) IM = 1... グラフィック表示領域 IM = 0	
		P _{n4}	*	IM	← SL _{nH} →							
CSRFORM	カーソル形状などの指定	C	0	1	0	0	1	0	1	1	L/R: 1 行中の表示ライン数(グラフィック・モード時...1) CS=1: カーソル表示あり BD=1: ブリンキングなし(CS=1なら常時点灯) CST: カーソル表示開始ライン値 BL: カーソル点滅周期 CFI: カーソル表示終了ライン値	
		P ₁	CS	0	0	← L/R →						
		P ₂	← BL _L →		BD	← CST →						
		P ₃	← CFI →				← BL _H →					
PITCH	映像メモリの水平方向ワード数の設定	C	0	1	0	0	0	1	1	1	} 水平方向のワード(16ビット)数	
P	← P →											
LPEN	ライトペン・アドレスの検出 読み出し専用	C	1	1	0	0	0	0	0	0	} LAD: ライトペン・アドレス	
		P ₁	← LAD _L →				← LAD _M →					
		P ₂	← LAD _M →				← LAD _H →					
		P ₃	← LAD _H →									
VECTW	描画に必要なパラメータの設定	C	0	1	0	0	1	1	0	0	} L=1: 直線描画 T=1: 傾斜しないグラフィックス文字描画 C=1: 円および円弧の描画 R=1: 四辺形描画 SL=1: 傾斜したグラフィックス文字(T=1) DGD=1: グラフィック描画(文字/グラフィック・モード)	
		P ₁	SL	R	C	T	L	← DIR →				
		P ₂	← DC _L →									
	P ₃	0	DGD	← DC _H →				← DC _H →				
	同様の順序で D ₂ 、D ₁ 、D ₀ を送出	P ₄	← DL →									
		P ₅	← DH →									
VECTE	グラフィックス描画開始	C	0	1	1	0	1	1	0	0		
TEXTW	グラフィックまたはテキスト・コード設定	C	0	1	1	1	1	← RA →				} RA: データを書き始めるラスト・アドレス PTN: グラフィック描画時の線のビット・パターン TX _n : ドット構成データ
		P ₁	← TX _n または PTN _L →									
		P ₂	← TX _n または PTN _H →									
		⋮	⋮									
		P _n	← TX _n →									
TEXTE	グラフィック/文字描画の実行開始	C	0	1	1	0	1	0	0	0		
CSRW	描画アドレス設定 文字描画時...P ₁ 、P ₂ のみ 文字モード時...EADMの上位3ビット0 文字/グラフィック・モード EADH=0	C	0	1	0	0	1	0	0	1	} EAD: 描画開始ワード・アドレス dAD: 描画開始ドット・アドレス	
		P ₁	← EAD _L →									
		P ₂	← EAD _M →									
		P ₃	← dAD →				0	0	← EAD _H →			
CSRR	描画アドレス読み出し 読み出し専用	C	1	1	1	0	0	0	0	0	} CSRW コマンドに同じ	
		P ₁	← EAD _L →									
		P ₂	← EAD _M →									
		P ₃	← EAD _H →									
		P ₄	← dAD _L →									
		P ₅	← dAD _H →									
MASK	マスク・レジスタ値の設定	C	0	1	0	0	1	0	1	0	} MASK: マスキング/ドット・アドレス・レジスタ値	
		P ₁	← MASK _L →									
		P ₂	← MASK _H →									

パラメータ	設定	機能
DE	0	表示停止
	1	表示開始
CHR, G	0 0	文字/グラフィック混在モード
	1 0	文字モード
	0 1	グラフィック・モード
	1 1	禁止
F	0	フラッシュ描画
	1	フラッシュレス描画
I, S	0 0	ノン・インターレース
	0 1	禁止
	1 0	インターレース
	1 1	インターレース・シュリンク
D	0	スタティック RAM 制御
	1	ダイナミック RAM 制御(リフレッシュ動作)
C/R		1行あたりの表示文字数設定(OOH~FFH=2~256文字/奇数)
HS		水平同期信号の幅の定義(OOH~1FH=1~32文字)
HFP		右方向の非表示区間の定義(OOH~3FH=1~64文字)
HBP		左方向の非表示区間の定義(OOH~3FH=1~64文字)
VSH+VSL		垂直同期信号の幅の定義(01H~1FH=1~31ライン)
VFP		下方向の非表示区間の定義(01H~3FH=1~63ライン)
VBP		上方向の非表示区間の定義(01H~3FH=1~63ライン)
L/FH+L/FL		1画面あたりの表示ライン数の設定 (OOOH~3FFH=1024, 1~1023ライン)

◀図 2-81▶

SYNC

図 2-82▶ PC9801 シリーズの
SYNC の設定値

パラメータ	専用 高解像度	標準ディ スプレイ
CHR	0	←
F	1	←
I	0	←
D	0	←
G	0	←
S	0	←
HS	07H	←
HBP	07H	ODH
HFP	09H	←
C/R	4EH	←
VS	08H	←
VBP	19H	25H
VFP	07H	0FH
L/F	190H	C8H

▶ SYNC

表示動作モード等の定義。パラメータで非常に多くの設定ができます。ごく簡単ですが図 2-81 に紹介しておきます。GDC コマンド一覧と併せて見てください。図 2-82 は PC98 シリーズにおける設定です。

▶ MASTER/SLAVE

マスタ動作かスレーブ動作の選択をします。
マスタ=6FH/スレーブ=6EH

■ GDC のコマンド(表示制御)

▶ START

表示開始の指示をします。

▶ STOP

表示停止の指示をします。

▶ ZOOM

表示時の拡大係数(ZR), グラフィック文字の拡大係数(ZW)の設定です。

ZR/ZW : 00H~0FH=1~16 倍

▶ SCROLL

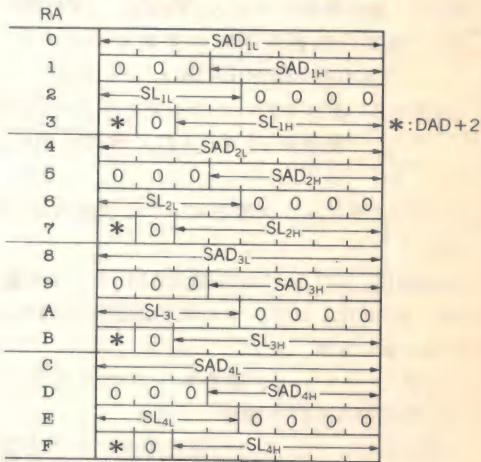
表示開始アドレス、画面分割表示領域の大きさの設定です。パラメータは GDC 内蔵の RAM に書いて、SCROLL コマンド実行時は RA(RAM アドレス)を設定するだけです。以下は、内蔵 RAM に送る各パラメータです。

RA : 変更する内部 RAM の先頭アドレス(0~15)

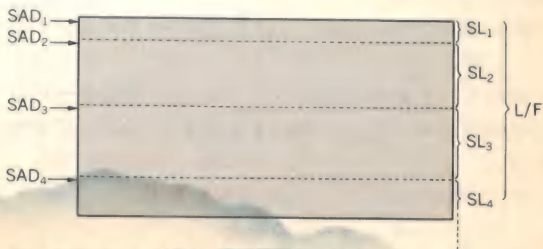
(1) 文字モードの場合

図 2-83 (a)に SCROLL コマンドの内蔵 RAM マップを、図 2-83 (b)に SCROLL コマンドの表示画面との対比を示します。

図 2-83▶ 文字モードのスクロール・コマンド

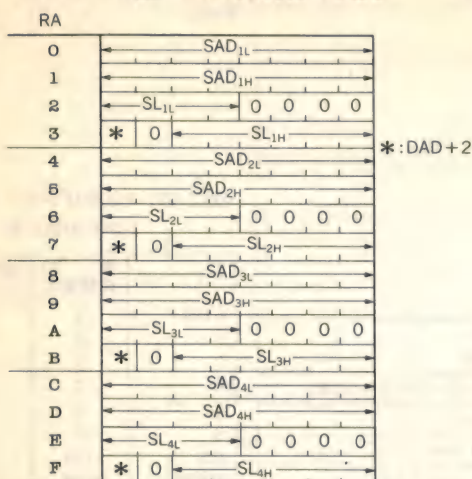


(a) 内蔵 RAM マップ

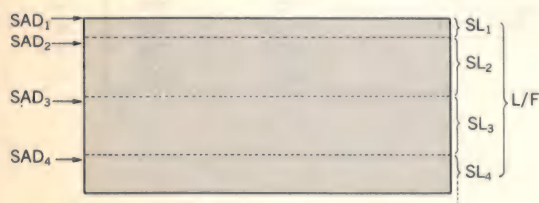


(b) 表示画面との対応

〈図 2-84〉 混在モードで文字表示のみ行う



(a) 内蔵 RAM マップ



(b) 表示画面との対応

SAD：表示開始アドレス(0000H～1FFFH)

SL：画面分割表示領域の大きさを示すライン数
(000H～3FFH=1024, 1～1023)

DAD+2：表示アドレスのインクリメント形態の定義
(0=+1/1=+2)。通常は“0”にして
おく。

(2) 文字/グラフィック混在モードで文字表示のみ行う
場合

SCROLL コマンドの内蔵 RAM マップを図 2-84

(a)に、SCROLL コマンドの表示画面との対比を図 2-84 (b)に示します。

(3) 文字/グラフィック混在モードでグラフィック表示/描画動作を行う場合

SCROLL コマンドの内蔵 RAM マップを図 2-85

(a)に、SCROLL コマンドの表示画面との対比を図 2-85 (b)に示します。

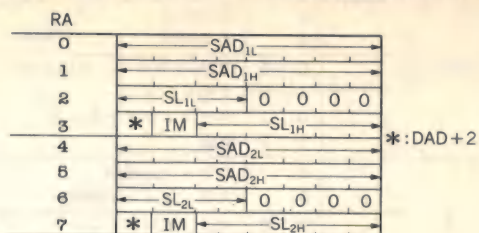
IM：表示アドレスのインクリメント・タイミング定義

文字モードまたは文字/グラフィック混在モードの文字領域においては、IM=1を設定することができません。GDC クロックが 2.5 MHz の場合は“0”，5 MHz の場合には“1”にして使用します。

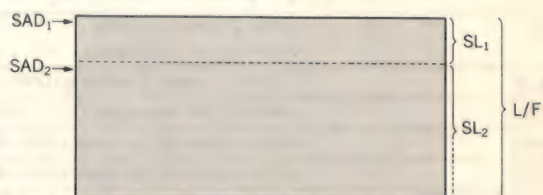
(4) グラフィック・モードの場合

SCROLL コマンドの内蔵 RAM マップを図 2-86

〈図 2-85〉 混在モードでグラフィック表示/描画動作を行う

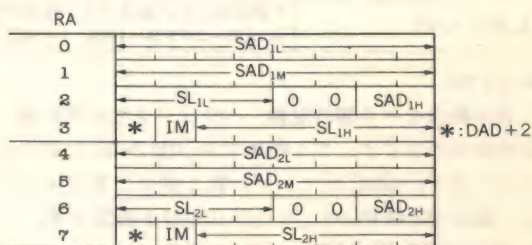


(a) 内蔵 RAM マップ

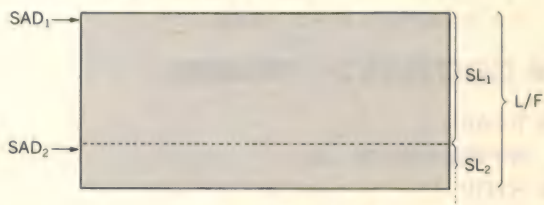


(b) 表示画面との対応

〈図 2-86〉 グラフィック・モード



(a) 内蔵 RAM マップ



(b) 表示画面との対応

(a)に、SCROLL コマンドの表示画面との対比を図 2-86 (b)に示します。

► CSRFORM

ライン・カウントの上限、文字表示時のカーソルの表示形態を定義します。

L/R：1 行中のライン数を定義する

(00H～1FH=1～32 ライン)

CS：カーソルの有無を定義する

(0=カーソルなし/1=カーソルあり)

BD：カーソルの点滅の有無を定義する

(0=点滅する/1=常時点灯)

BL：カーソルの点滅周期の定義

(01H～3FH=4～124)

CST：カーソルの表示開始ラインの定義

(00H～1FH=1～32 ライン)

〈図 2-87〉 VECTW

DIR	直線 (1)	円弧	四辺形
0			
1			
2			
3			
4			
5			
6			
7			

● : 描画始点 → : 第1方向 ▨ : 定義域
 → : 描画方向 - - - : 第2方向
 △ : 描画終了時の EAD 指示点

CF₁ : カーソルの表示終了ラインの定義
 (OOH~1FH=1~32 ライン)

▶ PITCH

映像メモリの横幅を定義します。これは表示領域 (C/R) とは別で、表示されない部分を含めた映像メモリの横幅です。スクロール等で使用します。SYNC コマンドを設定すると C/R の値で初期化されてしまうので、SYNC コマンドの後に使用します。

P₁ : OOH~FFH=0~255 (文字数/行)

▶ LPEN

ライト・ペンで得られたアドレスを出力します。出力は3バイトです。

D₁ : LAD_L (8 ビット)

D₂ : LAD_M (8 ビット)

D₃ : LAD_H (2 ビット)

◆ GDC のコマンド (描画制御)

主にグラフィック画面の描画用のコマンドです。比

〈図 2-88〉 描画時のアドレスの進み方

	DC	D	D ₂	D ₁	DM
初期値	0	8	8	-1	-1
直線	ΔX	2 ΔY - ΔX	2 ΔY -2 ΔX	2 ΔY	-
円	(r/√2) ↑	r-1	2(r-1)	-1	0
弧	N	r-1	2(r-1)	-1	M
四辺形	3	A*	B*	-1	A*

ΔX : X 座標変位, r : 半径, M : マスキング・ドット数,
 ΔY : Y 座標変位, N : 描画総ドット数, ↑ : 切り上げ,
 A* : 第1描画方向変位数, B* : 第2描画方向変位数

較的高機能的な描画コマンドを持っていますが、最近では CPU が高速化され、GDC で描画するより CPU で描画したほうが速い場合もあります。

▶ VECTW

描画方向 (8 方向), 描画種類 (直線/円弧/四辺形/文字), 描画用パラメータ (XY 軸/半径) 等を定義します。円は一度に 1/8 しか描画されません。

SL, R, C, T, L : 描画種類

直線 = 0, 0, 0, 0, 1

円弧 = 0, 0, 1, 0, 0

四辺形 = 0, 1, 0, 0, 0

DIR : 描画方向 [下向きが「0」で、左回りに 45 度単位で 8 方向 (図 2-87)]

DGD : 描画時のアドレスの進み方 (GDC クロック : 2.5 MHz = 0/5 MHz = 1)

DC, D, D₂, D₁, DM : 描画パラメータ (図 2-88)

参考に描画制御プログラム GDC.C をリスト 2-6 に示します。

▶ VECTE

直線, 四辺形, 円弧, 1 ドット描画の実行開始を指示します。

▶ TEXTW

実線, 破線等の線種データ, グラフィック文字用のドット構成データを設定します。

これらのデータは内部 RAM に送ります。

RA : 内部 RAM の先頭アドレス (0~7)

線種データは 2 バイト, グラフィック文字用のドット・データは 8 バイトです。

▶ TEXTE

グラフィック文字描画の実行開始を指示します。

▶ CSRW

描画実行ワード・アドレス (EAD) や, 描画実行ドット・アドレス (dAD) を設定します。文字制御時にはカーソル位置を, グラフィック制御時には描画開始点を定義します。

▶ SCRR

描画実行ワード・アドレス (EAD) や, 描画実行ドット・アドレス (dAD) を読み出します。読み出されるデータは 5 バイトです。

▶ MASK

MASK レジスタの値を設定します。

◆ GDC のコマンド(メモリ制御)

▶ WRITE

ドット修正モードや、映像メモリにデータ(このコマンドの後に続くパラメータ)を書き込むための準備をします。WRITE コマンドを実行する前に、書き込みみたいブロック(WLH による)数-1 と、書き込み開始アドレスを、VECTW, CSRW コマンドによって設定します。ドット修正モードだけ変更したい場合は必要ありません。

WLH: 00=ワード転送

01=禁止

10=下位バイト転送(上位バイトは0)

11=上位バイト転送(下位バイトは0)

MOD: ドット修正モードの選択

00=REPLACE(上書)

01=COMPLEMENT(反転)

10=CLEAR(0 を描く)

11=SET(1 を描く)

▶ READ

映像メモリの内容を読み出すコマンドです。あらかじめ読み込みみたいブロック(WLH による)数-1 と読み込み開始アドレスを、VECTW, CSRW コマンドによって設定します。

WLH: WRITE コマンドと同じ

MOD: WRITE コマンドと同じ

▶ DMAW

DMA を使用した WRITE コマンドです。使用方法は WRITE コマンドと同じです。PC98 では DMA は使用できません。

▶ DMAR

DMA を使用した READ コマンドです。使用方法は READ コマンドと同じです。PC98 では DMA は使用できません。

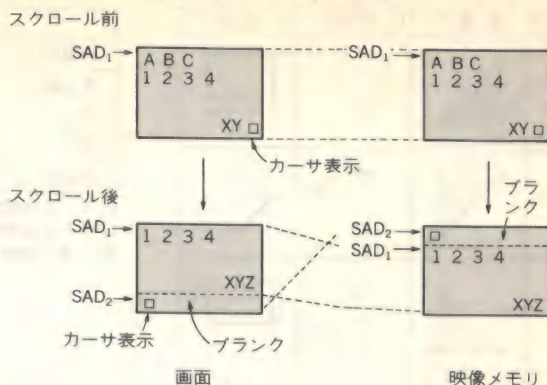
◆ スクロール、画面分割の方法

SCROLL コマンドによって設定する SAD(表示開始アドレス), SL(表示領域のライン数)により、表示画面を最大4個(グラフィック画面は2個)まで分割して表示できます。

SAD, SL は、SCROLL コマンドによって、GDC 内蔵 RAM にいったん格納されます。GDC が画面出力の際に、この RAM を見ながら出力します。

SAD, SL を少しずつ変化させることで、表示位置を少しずつ変えてスクロールさせることができます。例えば SAD₁=1 行, SL₁=25 行で画面全体が表示されているときに、1 行スクロールすると最上行は消え、最下位行に新しい行が見えてきます。ここで SAD₁を

〈図 2-89〉 ページ内スクロール



1 行繰り上げて SAD₁=2 とし、SL₁ も 1 行減るので SL₁=24 とすることで、1 行スクロールします。

これでは最下行が空白になってしまうので、前に SAD₁ で使用していた 1 行目のアドレスを SAD₂ で新たな表示画面として、SAD₂=2, SL₂=1 とし最下行に表示します。そして、スクロールを重ねるたびに SL₁ は減っていき、SL₂ が増えていきます(図 2-89)。

画面のスクロールは、テキスト画面だけでなく、グラフィック画面でも可能です。特にグラフィック画面は、ソフトウェアでスクロールさせると時間がかかりますが、SCROLL コマンドを使えば一瞬でスクロールが完了するので便利です。リスト 2-7 に示した SCROLL.C は、SCROLL コマンドのみを使用したグラフィックを画面スクロールするプログラムです。

◆ クロック周波数の設定

GDC に与えるクロック周波数は 1 水平走査期間内の表示時間 C/R と、そのときの映像メモリ・アクセス数(水平方向表示ドット数/1 ワード当たりのビット数)によって決定されます。

計算方法は以下のとおりになります。

HFP+HS+HBP+C/R=水平走査期間

C/R/16=1 回のメモリ・アクセスに要する時間

HS=水平同期信号の幅

HFP=右方向の非表示区間

HBP=左方向の非表示区間

C/R=1 行あたりの表示時間

表示サイクルは DAD+2 モードを使用しない通常表示では 2 クロックですので、1 回のメモリ・アクセスに要する時間の半分のコックを GDC に入力します。

PC98 のノーマル・モードで、専用高解像度 CRT (400 ライン) の水平走査周波数は、24.83 kHz (40.28 μ s) です。GDC に与えるパラメータは、HFP=07H (3.04 μ s), HS=07H (3.04 μ s), HBP=09H (3.8 μ s) となり、水平方向ドット数は 640 ドットで、その表示

＜リスト 2-7＞
スクロール・コマンド

```

/*
**  GDC (SCROLL) のテスト
**
**  グラフィック画面の垂直方向1ドットスクロール
**
**/

#include <dos.h>

#define      GDC_STAT      0xa0
#define      GDC_CMD       0xa2
#define      GDC_PARA      0xa0

void main(void)
{
    int      i, j, im;

    im = (peekb(0, 0x054d) & 0x04) << 4;      /* GDCクロックで IMを設定 */

    for (i = 1; i < 400; i++) {
        while ((inportb(GDC_STAT) & 0x04) == 0); /* FIFO が空くまで待つ */
        outportb(GDC_CMD, 0x70);                /* スクロールコマンド */
        outportb(GDC_PARA, (i*40) & 0xff);        /* SAD1 LOW */
        outportb(GDC_PARA, (i*40) >> 8);          /* SAD1 HI */
        outportb(GDC_PARA, ((400-i) << 4) & 0xff); /* SL1 LOW */
        outportb(GDC_PARA, im | ((400-i) >> 4));  /* DAD=0, SL1 HI */
        outportb(GDC_PARA, 0);                   /* SAD2 LOW */
        outportb(GDC_PARA, 0);                   /* SAD2 HI */
        outportb(GDC_PARA, (i << 4) & 0xff);      /* SL2 LOW */
        outportb(GDC_PARA, im | (i >> 4));        /* DAD=0, SL1 HI */
        for (j = 0; j < 30000; j++)
    }
}

```

＜図 2-90＞
ディスプレイ・タイミング

●ノーマル・モード

Symbol	専用高解像度 ディスプレイ	標準ディスプレイ
H	40.28 μ s (24.83 kHz)	62.58 μ s (15.98 kHz)
HDISP	30.4 μ s	44.70 μ s
HFP	3.04 μ s	4.47 μ s
HS	3.04 μ s	4.47 μ s
HBP	3.8 μ s	8.94 μ s
V	17.72 ms (440 H, 56.4 Hz)	16.33 ms (261 H, 61.2 Hz)
VDISP	16.11 ms (400 H)	12.52 ms (200 H)
VFP	0.28 ms (7 H)	0.94 ms (15 H)
VS	0.32 ms (8 H)	0.50 ms (8 H)
VBP	1.01 ms (25 H)	2.38 ms (38 H)

時間は $C/R = 4EH \cdot HDISP = 30.4 \mu s$ です。これからメモリ・アクセス時間は $0.76 \mu s$ で、GDCのクロックは 2.63 MHz (通称 2.5 MHz) となります。標準 CRT (200 ライン) では 1.79 MHz のクロックがかかります。

GDC の SYNC コマンドでは、これらのパラメータを定義してタイミングを作ることができます (SYNC コマンドのパラメータは時間指定ではなく文字数で指定する)。専用高解像度モードであっても、 640×400 ドット以外の画面を出せますが、GDC に与えられているクロックは固定されていますので、自由度は少なくなっています。

図 2-90 にディスプレイ・タイミングを示します。

◆ GDC のクロック

PC9801 シリーズでは、GDC は 2.5 MHz で動作するのが基本になっていますが、 5 MHz に切り替えることも可能です。GDC のクロックを倍 (IMAGE モード) にすることで描画速度は速くなりますが、GDC に

●ハイレゾリューション・モード

Symbol	タイミング	
	奇数フィールド	偶数フィールド
H	30.45 μ s (32.84 kHz)	
HDISP	23.41 μ s	
HFP	2.34 μ s	
HS	1.76 μ s	
HBP	2.93 μ s	
V	12.5 ms (410.5 H, 80 Hz)	
VDISP	11.42 ms (375 H)	
VFP	0.244 ms (8 H)	0.259 ms (8.5 H)
VS	0.152 ms (5 H)	
VBP	0.685 ms (22.5 H)	0.670 ms (22 H)

対して送るコマンドが若干異なってきます。反対に、計算上必要なクロックの半分を GDC に与えるモード (DAD+2 モード) もありますが使用しません。これらは SCROLL コマンドで設定します。

GDC の動作クロックは、システム共通領域 (0000 : 054DH・ビット 3) が “1” のときに 5 MHz 、 “0” のときに 2.5 MHz になります。 5 MHz 時には、SCROLL コマンドでは $IM=1$ 、VECTW コマンドでは $DGD=1$ にして使用します。

◆ GDC のバージョンの違い

PC98 シリーズでは、
 μ PD7220 PC9801FA 以前の機種
 μ PD7220A PC9801US
 μ PD72020 PC9801FA 以降の機種

のように、その発売時期によって新しい GDC が使用されています。これらは上位互換で、新しくなるたびに少しずつ機能が増えていきます。


```

/* GDCテストプログラム
**
#include <dos.h>

#define GDC_STAT 0xa0
#define GDC_CMD 0xa2
#define GDC_PARA 0xa0

#define VECTW 0x4c
#define VECTE 0x6c
#define CSRW 0x49
#define TEXTW 0x78
#define START 0x6b

#define VRAM_ON 0
#define VRAM_OFF 1
#define BIOS_CRT 0x18

typedef struct {
    int
    unsigned int dc;
    unsigned int d;
    unsigned int d2;
    unsigned int d1;
    unsigned int dx;
} vectpara;

int gdclock;

void setlinestyle(unsigned int pattern);
void line(int x1,int y1,int x2,int y2,int plane);
void box(int x1,int y1, int x2,int y2,int plane);
void circle(int x,int y,int radius,int plane);

void godraw(int x,int y,int plane,vectpara *vp);
void outgdccmd(int data);
void outgdcpars(unsigned int data,int len);
void outgdc(int port,int data);
void wait(void);

void crtinit(void);
int main()
{
    int i;

    crtinit();
    setlinestyle(0xfffff);
    for (i = 10; i < 640; i+=20) {
        line(320,200,i,0,0);
        line(320,200,i,399,0);
    }
    for (i = 10; i < 400; i+=20) {
        line(320,200,0,i,0);
        line(320,200,639,i,0);
    }
    for (i = 0; i < 150; i+=10) {
        box(i,i,639-i,399-i,1);
    }
    for (i = 1; i < 150; i+=5) {

```

```

        circle(320,200,i,2);
    }
    return 0;
}

/* 描画する線種を設定する。描画モードはリブレイス
**
void setlinestyle(unsigned int pattern)
{
    outgdccmd(0x20); /* WRITE WLH=00(ワード転送) MOD=00(REPLACE) */
    outgdccmd(TEXTW); /* TEXTW 線種データの設定 */
    outgdcpars(pattern,2);
}

/* (x1,y1) から (x2,y2) を結ぶ直線を plane の VRAMアレーンに描画する
**
void line(int x1,int y1,int x2,int y2,int plane)
{
    int dir,dx,dy;
    vectpara vp;

    /* 描画方向(dir)が0~3になるように座標を交換する */
    if ((x1 > x2) || ((x1 == x2) && (y1 > y2))) {
        dx = x1; x1 = x2; x2 = dx;
        dy = y1; y1 = y2; y2 = dy;
    }
    dx = x2-x1; dy = y2-y1;

    /* 座標から dir,dx,dy を求める */
    if (dy > 0) {
        if (dx < dy) {
            dir = 0;
            dx = dx; dy = dy;
        } else {
            dir = 1;
        }
    } else {
        dy = -dy;
        if (dx > dy) {
            dir = 2;
        } else {
            dir = 3;
            dx = dx; dy = dy;
        }
    }
    /* dir=3 は dx dy を交換する */

    vp.dir=dir;
    vp.dc = dx; vp.dl = 2*dy; vp.d = vp.dl-dx; vp.d2 = vp.d-dx; vp.dm = 0;
    godraw(x1,y1,plane,&vp);
}

/* (x1,y1)を左上、(x2,y2)を右下とする長方形を plane のVRAMに描画する。
**
void box(int x1,int y1, int x2,int y2,int plane)
{
    int dx,dy,dx2,dy2;
    vectpara vp;

    /* 左上が(x1,y1) 右下が(x2,y2)になるように座標を交換する。 */
    if (x1 > x2) {

```

```

*/
void outgdccmd(int dat)
{
    outgdc(GDC_CMD, dat);
}

/* GDC にパラメータを送る。 len 1=BYTE 2=WORD
*/
void outgdcpa(unsigned int dat, int len)
{
    int i;
    for (i = 0; i < len; i++) {
        outgdc(GDC_PARA, dat << 0xff);
    }
}

/* G-GDC に対して 1 バイトのデータを出力
*/
void outgdc(int portadr, int dat)
{
    wait();
    while ((inportb(GDC_STAT) & 0x02) != 0) wait();
    outportb(portadr, dat);
}

/* WAIT
*/
void wait(void)
{
    outportb(0x5f, 0);
    outportb(0x5f, 0);
}

/* 8色 400ライン
*/
/* 描画 V RAM 表 表示 V RAM 表
*/
void crtinit(void)
{
    union REGS regs;
    regs.h.ah = 0x42;
    regs.h.ch = 0xc0;

    int86(BIOS_CRT, &regs, &regs);
    int86(BIOS_CRT, &regs, &regs);

    outports(0x5a, 0); /* 8色モード */
    wait();
    outports(0xa4, 0); /* 表示画面 V RAM 表 */
    wait();
    outports(0xa6, 0); /* 描画画面 V RAM 表 */
    wait();
    outports(GDC_CMD, START); /* 表示開始 */

    gdeclock = (peekb(0, 0x054d) & 0x04) >> 2; /* gdeclock 1=5M 0=2.5M */
}

```

```

    day = x1; x1 = x2; x2 = day;
}
if (y1 > y2) {
    day = y1; y1 = y2; y2 = day;
}
dx = x2-x1; dy = y2-y1;

dir = 0; /* DIRは0固定 */
vp.srctdir = 0x40|dir; /* 四辺形 */
vp.dc = 3; vp.dl = -1; vp.d = dy; vp.d2 = dx; vp.dm = dy;

godraw(x1, y1, plane, &vp);
}

/* 中心座標(x,y) 半径 radius の円を plane の V RAM に描画する。
*/
void circle(int x, int y, int radius, int plane)
{
    int dir;
    vectepara vp;
    int xx[8], yy[8];

    /* DIR の変化による描画開始点を設定 */
    xx[0] = xx[3] = x-radius; yy[0] = yy[3] = y;
    xx[1] = xx[6] = x; yy[1] = yy[6] = y-radius;
    xx[2] = xx[5] = x; yy[2] = yy[5] = y-radius;
    xx[4] = xx[7] = x-radius; yy[4] = yy[7] = y;

    vp.dc = (int)((long)radius*10000/14142)+1;
    vp.d = radius-1; vp.d2 = 2*vp.d; vp.dl = -1; vp.dm = 0;

    for (dir = 0; dir < 8; dir++) {
        vp.srctdir = 0x20|dir;
        godraw(xx[dir], yy[dir], plane, &vp);
    }
}

/* VECTW, CSRW, VECTE
*/
void godraw(int x, int y, int plane, vectepara *vp)
{
    unsigned int ead;

    outgdccmd(CSRW);
    ead = (unsigned int)((plane+1)&4)*0x4000+y*40*x/16;
    outgdcpa(ead, 2);
    outgdcpa((xx[6]<<4, 1);

    outgdccmd(VECTW);
    outgdcpa(vp->srctdir, 1);
    outgdcpa((vp->dc) | (gdeclock < 14), 2); /* DGD=1:5MHz = 0.2.5MHz */
    outgdcpa(vp->d2, 2);
    outgdcpa(vp->d1, 2);
    outgdcpa(vp->dm, 2);

    outgdccmd(VECTE);
    while ((inportb(GDC_STAT) & 0x08) != 0) wait(); /* 描画終了まで待つ */
}

/* GDC にコマンドを設定する。
*/

```


フロッピー・ディスク・インターフェース

▶使用 LSI

μ PD765A 相当

<1MB 用フロッピー・ディスク・インターフェース>

▶ I/O アドレス

90H, 92H (FDC)

94H (ライト・コントロール/リード・シグナル)

▶使用割り込み

IR₁₁

▶使用 DMA

2

<640KB 用フロッピー・ディスク・インターフェース>

▶ I/O アドレス

C8H, CAH (FDC)

CCH (ライト・コントロール/リード・シグナル)

▶使用割り込み

IR₁₀

▶使用 DMA

3

<640KB/1MB インターフェース切り替え>

▶ I/O アドレス

BEH

PC9801 で使用されるフロッピー・ディスク・ドライブには、8 インチ 2D (1MB), 5 インチ 2D (320KB), 5 インチ 2DD (640KB), 5 インチ 2HD (1MB), 5 インチ 640KB/1MB 両用型, 3.5 インチ 2DD (640KB), 3.5 インチ 2HD (1MB), 3.5 インチ 640KB/1MB 両用型が使われています。

フロッピー・ディスク・インターフェースには、320K, 640KB, 1MB, 本体に内蔵の 640KB/1MB 両用タイプがあります。また、最近の機種では PC/AT 互換機で標準の 3.5 インチ 2HD (1.44MB) のメディアを扱うこともできます。

◆ 5 インチ 2D (320KB) フロッピー・ディスク・インターフェース

5 インチ 2D 型は PC8001, PC8801 等で使用されていた外部接続用のインテリジェント型のフロッピー・ディスク・ドライブ・ユニット (PC80S31 等) をパラレル・インターフェース (8255) 経由でアクセスできましたが、PC9801M 以降の機種には、このインターフェースは付いていません。

しかし、640KB/1MB 両用型のドライブを持つ機種では、BIOS レベルで 2D を仮想的に読み書きすることが可能です。ただし、ドライブの構造的な問題で書き込みは保証されません。

◆ 640KB・1MB 専用フロッピー・ディスク・インターフェース

640KB および 1MB のフロッピー・ディスク・インターフェースには、FDC (フロッピー・ディスク・コントローラ) として μ PD765A の相当品 (以降 765 と略す) を使用しています。FDC のデータ転送には DMA を使用します。

640KB と 1MB では、使用する割り込みレベル、DMA チャンネル、I/O ポートが別になっていて、基本的には独立した FDC を持つ構造になっており、それぞれ 4 台ずつの FDD (フロッピー・ディスク・ドライブ) を付けることが可能です。

PC9801VM/UV 以降の機種にはすべて、640KB/1MB 両用型のフロッピー・ディスク・インターフェースが付いていますから、拡張スロットに 640KB, 1MB 専用のフロッピー・ディスク・インターフェースを付ける必要はなくなっています。別に専用インターフェースを付けたい場合は、本体の両用型インターフェースをどちらかのモードに固定して使うことになります。

◆ 640KB/1MB 両用型フロッピー・ディスク・インターフェース

640KB/1MB 両用型のインターフェースは、640KB, 1MB 用の FDC を共用しています。しかし、**割り込み、DMA、I/O アドレス等は、それぞれの専用インターフェース同様に独立して割り当てられていて、モード切り替えポート (OBEH) で切り替えることができます。**

通常、BIOS で使用する場合は、640KB 専用モードと、640KB/1MB 自動切り替えモードがあり、ディップ・スイッチ (SW₃₋₁/SW₃₋₂) で切り替えます。640KB 専用モードでは、640KB 専用インターフェースと同様の DMA チャンネル (#3), 割り込みレベル (IR₁₀) を使用し、640KB/1MB 自動切り替えモードでは、1MB 専用インターフェースと同様の DMA チャンネル (#2), 割り込みレベル (IR₁₁) を使用します。

640KB/1MB 両用のタイプの FDD, フロッピー・ディスク・インターフェースが搭載されている機種でも、仕様では、外部に増設する FDD は 1MB 専用になっています。これは、増設用 FDD コネクタに 640KB/1MB 切り替え出力が出ていないせいです。増設用 FDD に 640KB/1MB 両用型を使い、増設用 FDD ユニットの 2 ピンと、内蔵 FDD ユニットの 2 ピンを接続すれば、増設用 FDD も 640KB/1MB の切り替えができます (ただし、MS-DOS の一部のコマンドは自動切り替えに対応できない)。

また、FDD のストラップを変更し FDD 単体で自動切り替えできるモードにし、フロッピー・ディスク・

〈図 2-91〉 モード・チェンジ・レジスタ

命 令	I/O ポート ・アドレス	R/ W	データ								備 考
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
ライト・モード・チェンジ	OOBE	W	0	0	0	0	0	EMTON	FDD EXC	PORT EXC	μPD765A 外部レジ スタ、初期値は 00
リード・モード・ステータス	OOBE	R	×	×	×	×	DSW	FIX	FDD EXC	PORT EXC	

BIOS に細工をすることで 640KB/1MB 自動切り替えにすることも可能です。フリー・ソフトウェアでは「DUALDRV.SYS」(T. Haraikawa 氏製作)等があります。

◆ モード・チェンジ・レジスタ

640KB/1MB 両用型フロッピー・ディスク・インターフェースの切り替えを行うポートです。PORT EXC ではインターフェースの I/O アドレス、DMA チャンネル、割り込みレベル等の CPU に密接する部分の切り替えを行います。FDD EXC は、FDC の動作モードを変えるもので、VFO パラメータや FDC クロック等の切り替えを行います。

640KB/1MB 自動切り替えモードでは、PORT EXC をどちらかに固定し、FDD EXC を実際に読める(エラーが起きない)ほうに切り替えることで、両方のメディアを扱っています。通常では PORT EXC は電源投入時の初期設定で固定化され、動作途中に変更されることはありません(図 2-91)。

▶ D₀: PORT EXC

インターフェースの I/O アドレスを設定します。

1: 1MB インターフェース用(90H, 92H, 94H)

0: 640KB インターフェース用(C8H, CBH, CCH)

▶ D₁: FDD EXC

VFO パラメータと FDC のクロックを設定します。

1: 1MB モード

0: 640KB モード

▶ D₂: EMTON

FDD のモータ制御用。EMTON=1 で 1MB モード時に MTON の設定を有効にします。EMTON=0 で 1MB モード時に強制的に、常にモータ ON の状態にします。

◆ モード・チェンジ・レジスタ(BEH/リード)

▶ D₀: PORT EXC

インターフェースの I/O アドレス設定状態を読み出します。

1: 1MB インターフェース用(90H, 92H, 94H)

0: 640KB インターフェース用(C8H, CBH, CCH)

▶ D₁: FDD EXC

VFO パラメータと FDC のクロック設定状態を読み出します。

1: 1MB モード

0: 640KB モード

▶ D₂: FIX

ディップ・スイッチ SW₃₋₁の状態を示します。

1: PORT EXC を無効とする固定モード

0: 有効とする自動切替えモード

▶ D₃: DSW

ディップ・スイッチ SW₃₋₂の状態を示します。

FIX=0 の場合に FDD の立ち上がり状態を読み取り、FIX=1 の場合に固定するモードを示します。

1: 1MB モード

0: 640KB モード

▶ D₆: TMF(EPSON-PC シリーズ)

1MB インターフェース使用時

0: FDD モータ停止

1: FDD モータ回転中

640KB インターフェース使用時

0: ワンショット・タイマ動作中

1: ワンショット・タイマ、タイム・アウト

▶ D₇: IFDC(EPSON-PC シリーズ)

0: FDC の割り込みがある

1: FDC の割り込みがない

◆ μPD765A(FDC)

FDC には二つの I/O アドレスがあり、ステータス・レジスタもリードとデータ・レジスタの読み書きがあります。データ・レジスタはコマンド、パラメータ、データ、リザルト・ステータスがあります。

● ステータス・レジスタ(90H, C8H/リード)

▶ ステータス読み出し

ステータス・レジスタではドライブの状態、FDC の状態等が読み出せます。

● データ・レジスタ(92H, CBH/リード&ライト)

▶ コマンド設定(Command Phase)

「コマンド」の設定は、FDC がアイドルリング状態のときにデータ・レジスタに書き込むことで実行します。パラメータが必要な場合はコマンドに続いてデータ・レジスタに書き込みます。

▶ コマンド実行(Execution Phase)

必要なパラメータの送出が終わったら、FDC は Execution Phase に入り、与えられたコマンドを実行します。必要なら DMA 等を通してメモリに対して

〈図 2-92〉 ライト・コントロールの I/O アドレス

	命 令	I/O ポート ・アドレス	R/ W	データ								備 考
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
FDC μPD765	ライト・コマ ンド	92	W	← コマンド・レジスタ →								μPD765A へコマンドを セットする
		CA										
	ライト・レジ スタ	92	W	← パラメータ →								μPD765A へのパラメー タをセットする。NON- DMA モード時は FD へ の書き込みデータもセッ トする
		CA										
	リード・ステ ータス	90	R	← ステータス・レジスタ →								μPD765A からステータ スを引き取る
		CB										
リード・デー タ	92	R	← リザルト・ステータス →								μPD765A からリザル ト・ステータスを引き取 る。NON-DMA モード 時には FD から読み取っ たデータも引き取る	
	CA											
コントローラ	ライト・コン トロール	94	W	RST	FRY	×	×	×	×	×	×	
		CC		RST	FRY	AIE	DMAE	MTON	TMSK	×	TTRG	
	リード・スイ ッチ/シグナ ル	94	R	FINT ₁	FINT ₀	DMACH	0	TYP ₁	TYP ₀	0	0	TYPE ₁ FDD #3/#4 TYPE ₀ FDD #1/#2
		CC		FINT ₁	FINT ₀	DMACH	RDY	TYP ₁	TYP ₀	0	0	0 : 1 MB FDD 1 : 両用 FDD

上段 1MB インターフェース、下段 640KB インターフェース

PC9801VF2/VM0,2,4/UV2/VM21/VX0,2,4/UV21/VX01,21,41 では、EMTON ビットは無効

読み書きします。

▶ 結果出力 (Result Phase)

最後は、コマンドの実行結果を報告するためのリザルト・ステータス情報をセットし、CPU がそれらの情報をデータ・レジスタを通して読み出します。必ず INT 要求によって処理されますので、すべてのリザルト・ステータス、パラメータ情報を指定された順序に従って読み取る必要があります。

■ ライト・コントロール(94H, CCH/ライト)

94H=1MB インターフェース用、
CCH=640KB インターフェース用

図 2-90 に I/O アドレスを示します。

▶ D₀ : TTRG (640KB のみ)

VFO の TRIG IN 端子の入力信号であり、ディスク・ドライブのモータ ON/OFF 制御の時間設定用タイマのトリガとして使用します。TMSK=1, TTRG=1 とすると、約 100 ms 後、割り込み信号が発生します。このタイマ機能は、トリガ入力後、100 ms 以内に再びトリガを入力すれば、後のトリガが有効になります。

EPSON-PC シリーズでは 1MB でも有効ですので、

TTRG=0 にします。

▶ D₂ : TMSK (640KB のみ)

FDC からのタイマ割り込みをマスクするレジスタで、TMSK=0 で FDC からのタイマ割り込みが禁止されます。電源投入直後には「0」にしておきます。「1」にしてからタイマをトリガするとタイム・アウト以前に割り込み信号が 1 回出るので、注意する必要があります。

EPSON-PC シリーズでは 1MB でも有効ですので、TMSK=0 にします。

▶ D₃ : MTON (640KB のみ)

MTON=0 で FDD のモータを同時にすべて停止させます。

▶ D₄ : DMAE (PC9801 では無効)

DMA チャネルを使ってデータ転送を行うとき、DMA コントローラからの DRQ 信号、DACK 信号の許可/不許可を指示します。DMA 使用時は DMAE=1 とします。

▶ D₅ : AIE (640KB/1MB 両用型と PC9801U のみ)

AIE=0 で FRY の入力を無視し、前の状態のままにします。

▶ D₆ : FRY (640KB 専用にはなく、PC9801U にはある)

〈図 2-93〉 リード・シグナル

VF			VM 他		
TYP _i	TYP ₀	種 別	TYP _i	TYP ₀	種 別
0	0	外付け1MBのみ	1	0	内蔵両用
1	0	内蔵両用	0	1	外付け1MB
0	1	外付け1MB			

765 の READY 端子に、ディスク・ドライブの RDY 信号と論理和(AND)してつながっていて、ドライブの接続状態やドライブの電源投入状態をチェックするのに使用されます。

これらのチェックをするために、ドライブにキャリブレート動作をさせて、Track00 信号が返ってくるか調べますが、765 の READY 端子が OFF であると、キャリブレート動作がされず判別できないため、FRY=1 で強制的に READY 端子を ON にして調べます。通常動作のときは、FRY=0 としておかないと、Not Ready を検出できなくなります。

▶ D₇ : RST

765 の RESET 端子につながっていて、765 を初期化するのに使用します。初期化は 765 へのコマンド、パラメータの転送シーケンスや、リザルト・ステータス転送シーケンスが乱れたときに使用します。なお、初期化は電源投入直後や RESET スイッチを押したときに、ハードウェアで行われます。RST=1 で初期化されます。

◆ リード・シグナル(94H, CCH/リード)

94H=1MB インターフェース用、
CCH=640KB インターフェース用

▶ D₂~D₃ : TYP₀~TYP₁(640KB/1MB 両用型のみ)

使用できる 4 台の FDD の種別(640KB/1MB 両用型、1MB 専用型)を読み出せます(図 2-93)。

▶ D₄ : RDY(640KB のみ)

FDD の RDY 端子の状態を読み取ります。RDY=1 で Ready です。ドライブの有無のチェックは FRY の説明のように行います。このビットは、通常のシークや読み書きコマンドを実行する場合に、FDD の Ready をチェックするためのものです。

PC9801U を除く 640KB 専用機種、インターフェース・ボードは、765 の READY 端子は「1」で固定されています。

PC9801 の 94H・D₄は、プリンタ・インターフェースの PSTB に使用しています。他機種では PSTB は 37H にあります。

▶ D₅ : DMACH

640KB または 1MB の専用インターフェースでは、基板上のディップ・スイッチの内容が読み出せます。システム既定値や 640KB/1MB 両用機種では、1MB=0、640KB=1 が読み出せます。

▶ D₆~D₇ : FINT₀~FINT₁

640KB または 1MB の専用インターフェースでは、基板上のディップ・スイッチの内容が読み出せます。システム既定値や、640KB/1MB 両用機種では、FINT₀=1、FINT₁=0 が読み出せます。

ハード・ディスク・インターフェース

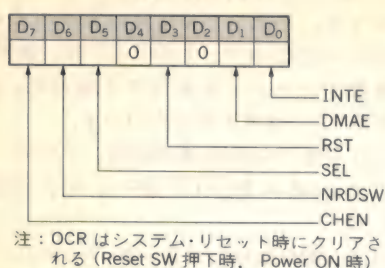
PC98 シリーズで使用されるハード・ディスク・ドライブ(HDD)には、SASI, SCSI, IDE の 3 種類があります。SASI は古くから PC98 シリーズに使用されてきた方式で、2 ドライブ・合計 80M バイトまでしかサポートされていません。SCSI は近年、SASI の代わりに盛んに使われるようになった方式で、7 台までのユニットを制御でき、CD-ROM ドライブや MO ドライブ、記憶装置以外のユニットも制御できます。

IDE は、PC/AT 互換機等で使用されている HDD で、インターフェースが HDD 内部に付いているために、基本的にはバスに直結する形をとります。また、

〈図 2-94〉 SASI インターフェースのアドレス

命 令	I/O ポート ・ アドレス	R/ W	データ								備 考
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
ODR アウトプット・データ・レジスタ	80	W	OD ₇	OD ₆	OD ₅	OD ₄	OD ₃	OD ₂	OD ₁	OD ₀	
IDR インプット・データ・レジスタ	80	R	ID ₇	ID ₆	ID ₅	ID ₄	ID ₃	ID ₂	ID ₁	ID ₀	
OCR アウトプット・コントロール・レジスタ	82	W	—	NRDSW	SEL	0	RST	0	DMAE	INTE	内蔵タイプ
			CHEN	NRDSW	SEL	0	RST	0	DMAE	INTE	外付けタイプ
ISR インプット・ステータス・レジスタ	82	R	REQ	0	BSY	MSG	CXD	IXO	—	INT	内蔵タイプ NRDSW≠1
			REQ	ACK	BSY	MSG	CXD	IXO	—	INT	外付けタイプ NRDSW=1
			CT ₁	CT ₀	DT ₀₂	DT ₀₁	DT ₀₀	DT ₁₂	DT ₁₁	DT ₁₀	NRDSW=0

〈図 2-95〉 OCR



ソフトウェア的には SASI と同等に扱えるようになっています。合計 80M バイトの容量制限はありませんが、ドライブ数は 2 台までです。

◆ SASIインターフェース(PC9801-27)

▶ I/O ポートアドレス

80H, 82H

▶ 割り込みレベル

INT₃(IR₉₁)

▶ DMA チャンネル

#0

SASI インターフェースは二つの I/O アドレスと、四つのレジスタから構成されています。インターフェースには特別な制御用シーケンスを持った LSI は使用しておらず、**基本的にはソフトウェアによって、HDD の制御シーケンス信号を作ります。**また、データの受け渡しは、98 内蔵の DMA により転送されます。

図 2-94 にアドレス一覧を示します。

▶ ODR/IDR(80H/ライト・リード)

HDD に対して送る制御用コマンド&データの受け渡しをします。データの受け渡しを DMA でなく CPU が直接行う場合は、このポートを使用します。

▶ OCR(82H/ライト)

SASI インターフェースの制御用ポートです。

OCR の内容を図 2-95 に示します。

D₀ : INTE……INTE=1 のときに割り込みの発生が許可されます。

D₁ : DMAE……DMAE=1 で DMA モードになります。

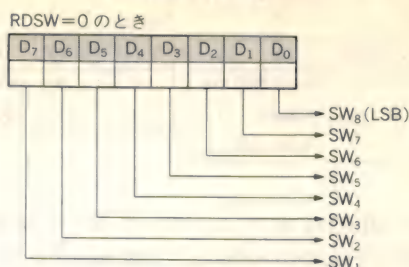
D₃ : RST……RST を 1 → 0 に変化させたときに、HDD に対して「RST」信号を送ります。

HDD のコントローラ (HDC) をリセットします。

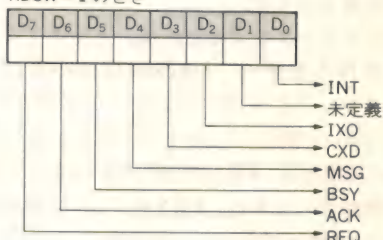
D₅ : SEL……SEL=1 のときに HDD の制御信号である「SEL」を ON にします。

コントローラを選択するとき ON にします。コントローラ番号 (01H) はデータ信号 (ODR) から送ります。

〈図 2-96〉 ISR



RDSW=1 のとき



す。

D₆ : NRDSW……ISR レジスタで読み取れる内容を切り替えます。NRDSW=0 で HDD インターフェース・ボード上のディップ・スイッチを読み出します。

NRDSW=1 で HDD からの制御信号を読み出します。
D₇ : CHEN……CHEN=1 で、インターフェースのデータ・制御線 (内部バス) の出力が許可されます。内蔵型インターフェースでは無効です。

▶ ISR(82H/リード)

図 2-96 に ISR を示します。

D₀ : INT……割り込み線の状態を読み出します。

INTE=0 の場合はリセット状態にされ、INT=1 で割り込み状態となります。

D₂ : IXO……HDD 制御用の入力で、内部バスの IXO の状態を示します。

データ転送の方向 (0=98 → HDD/1=HDD → 98)

D₃ : CXD……HDD 制御用の入力で、内部バスの CXD の状態を示します。

データ転送の情報 (0=データ情報/1=制御情報)

D₄ : MSG……HDD 制御用の入力で、内部バスの MSG の状態を示します。

動作完了状態で、ポスト・ステータス・バイトが送られている (MSG=1)

D₅ : BSY……HDD 制御用の入力で、内部バスの BSY の状態を示します。

内部バスが動作中 (BSY=1)

D₆ : ACK……HDD 制御用の入力で、内部バスの ACK の状態を示します。

データ転送 (1 バイトごと) の応答信号で、IDR/ODR をアクセスしたときは「1」、REQ 信号がオフに

〈図 2-97〉 WD33C93 の I/O アドレス

LSI	命 令	I/O ポート ・アドレス	R/ W	データ								備 考
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
WD 33 C93	REGISTER ADDRESS SER	CC0	W	AR ₇	AR ₆	AR ₅	AR ₄	AR ₃	AR ₂	AR ₁	AR ₀	CONTROL REGIS TER のアドレス・セ ット
	AUXILIARY STATUS	CC0	R	INT	LCI	BSY	CIP	0	0	PE	DBR	補助ステータス
	CONTROL REGISTER	CC2	R/ W	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	WD33C93 のコントロ ール・レジスタ
制御用 レジスタ	STATUS READ	CC4	R	—	TCI	—	—	—	—	DMA ₁	DMA ₀	DMA チャンネル・リー ド、 TC 割り込みのモニタ
	COMMAND WRITE	CC4	W	0	0	0	TCIR	TCMR	TCMS	DMER	DMES	DMA イネーブル、 TC マスク・セット/リ セット、 TC 割り込みリセット

なると“0”になります。

D₇: REQ……HDD 制御用の入力で、内部バスの REQ の状態を示します。

データ転送の要求信号 (REQ=1)

◆ SCSIインターフェース(PC9801-55)

▶使用 LSI

WD33C93/A

▶ I/O ポート・アドレス

CC0H, CC2H, CC4H (標準設定)

▶割り込みレベル

INT₃(IR₉₁) (標準設定)

▶ DMA チャンネル

#0 (標準設定)

SCSI インターフェースには多くの種類がありますが、もっともスタンダードな PC9801-55 ボードについて解説します。これは、SCSI コントローラに、WD33C93/WD33C93A (以下 33C93 と略す) を使用し、転送プロトコルの一部をインテリジェントに処理する LSI です。

図 2-97 のように、I/O ポート・アドレスは、標準では CC0H, CC2H, CC4H を使用しますが、これは基板上のジャンパで、CDOH~CD4H, CEOH~CE4H, CFOH~CF4H へ変更ができます。通常は CC0H~CC4H で使用します。

割り込みレベルについても、INT₃以外に変更可能です。SASI インターフェース等と共存する場合に、INT₃以外に設定します。DMA チャンネルも #0 以外に変更可能ですが、拡張スロットには #1 は出力されていませんし、80286 以上の CPU を持つ機種 of 拡張スロットには #2 も出力されていませんので、選択の幅

は多くありません。

他にも設定として、ローカル・メモリ・アドレスがあります。機種、CPU に合わせて変更するディップ・スイッチで、使用する拡張 ROM アドレスが変わります。

I/O アドレスは三つ割り当てられていますが、このうち二つは 33C93 が使用します。アドレスは 4 種類に可変できますが、ここでは CC0H~CC4H を例に挙げて解説します。

▶ CC4H: DMA チャンネル・リード、TC 割り込みモニタ/リード

D₀₋₁: DMA₀-DMA₁……ボードに搭載されている DMA チャンネルの切り替えのディップ・スイッチを読み出します。この設定に合わせて、使用する DMA チャンネルを決定します。

D₆: TCI……DMATC₀信号により割り込みのモニタをします。TCI=1 で割り込み発生

▶ CC4H: DMA イネーブル、TC マスク、TC 割り込み/ライト

D₀: DMES……DMA のイネーブル・セット

D₁: DMER……DMA のイネーブル・リセット

D₂: TCMS……TC 割り込みマスクのセット

D₃: TCMR……TC 割り込みマスクのリセット

D₄: TCIR……TC 割り込みのリセット

● WD33C93

WD33C93 は、SCSI コントローラというより、ディスク・コントローラの性格が強く、ヘッド、シリンダ、セクタ等の管理が LSI 単体でできます。また、コンビネーション・コマンドを持ち、割り込み回数を削減できます。

▶ CC0H: 33C93 のレジスタ・アドレス・セット/ライト

アクセスする 33C93 のレジスタのアドレス番号を

〈図 2-98 (a)〉 WD33C93 レジスタ構成

アドレス	R/W	レジスタ	ビット								アドレス (16進)	
			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
CC0	W	Address	アドレス								—	
	R	Auxiliary Status	INT	LCI	BSY	CIP	0	0	PE	DBR	—	
CC2	R/W	Own ID	0	0	0	0	0	ID ₂	ID ₁	ID ₀	00	
	R/W	Control	DMA	WDB	0	0	0	0	HA	HPE	01	
	R/W	Timeout Period	タイム・アウト定数								02	
	R/W	Total Sectors	セクタ数/トラック								03	
	R/W	Total Heads	ヘッド数								04	
	R/W	Total Cylinders(MSB)	シリンダ数(上位バイト)								05	
	R/W	Total Cylinders(LSB)	シリンダ数(下位バイト)								06	
	R/W	Logical Address(MSB)	論理セクタ(最上位バイト)								07	
	R/W	Logical address	論理セクタ								08	
	R/W	Logical adress	論理セクタ								09	
	R/W	Logical Address(LSB)	論理セクタ(最下位バイト)								0A	
	R/W	Sector Number	セクタ番号								0B	
	R/W	Head Number	ヘッド番号								0C	
	R/W	Cylinder Number(MSB)	シリンダ番号(上位バイト)								0D	
	R/W	Cylinder Number(LSB)	シリンダ番号(下位バイト)								0E	
	R/W	Target LUN	ターゲット論理ユニット番号								0F	
	R/W	Command Phase	0	コンビネーション・コマンドの終了状態								10
	R/W	Synchronous Transfer	0	転送サイクル			0	オフセット			11	
				TP ₂	TP ₁	TP ₀		OF ₂	OF ₁	OF ₀		
	R/W	Transfer Count(MSB)	転送バイト数(最上位ビット)								12	
	R/W	Transfer Count	転送バイト数								13	
	R/W	Transfer Count(LSB)	転送バイト数(最下位ビット)								14	
	R/W	Destination ID	0	0	0	0	0	DI ₂	DI ₁	DI ₀	15	
	R/W	Source ID	ER	ES	0	0	SIV	SI ₂	SI ₁	SI ₀	16	
	R	SCSI Status	SCSI ステータス								17	
	R/W	Command	SBT	コマンド・コード								18
	R/W	Data	入出力データ								19	
	R/W	Momory Bank	0	ROM ₀	0	0	MEM ₁	IRE ₁	WRS ₁	0	30	
	R	Memory Window	—	HST ₁	HST ₀	WND ₄	WND ₃	WND ₂	WND ₁	WND ₀	31	
	R/W	NEC リザーブ	使用禁止								32	
	W	NEC リザーブ	使用禁止								33	
	R	RESENT/INT	RRST	—	ILV ₂	ILV ₁	ILV ₀	ID ₂	ID ₁	ID ₀	33	
	R/W	NEC リザーブ	使用禁止								34	
	W	NEC リザーブ	使用禁止								35	

〈図 2-98 (b)〉 WD33C93 の I/O コマンド・ビット

ビット名	R/W	0/1	機 能	コマンド名
WRS ₁	W	0	SCSI インターフェース・ボードからの SCSI バス・リセットを解除する	Memory
		1	本ビット書き込み後 SCSI バス上 RST 信号が “L” アクティブになる	
	R	0 [☆]	SCSI バスが SCSI インターフェース・ボード側からはリセットされていないことを示す	
		1	SCSI の RST 信号を、SCSI インターフェース・ボードへの I/O コマンドによりアクティブにした、SCSI バスの他のデバイスからリセットされても本ビットは 1 にならない(リセットし続ける)	
IRE ₁	R/W	0 [☆]	SCSI インターフェース・ボードからシステム側への割り込みがマスクされる。ただし、割り込み要因自体を本ビットでクリアすることはできない	
		1	SCSI インターフェース・ボードからシステム側への割り込みが許可される。本ビットが “0” の間に割り込み要因が発生したときは、これを “1” にしてマスクを解除しても割り込みは発生しない。しかし、マスク解除前に割り込み要因をクリアすればこの限りではない	
MEM ₁	W	0	ローカル・メモリ (ROM) をシステム側からアクセスできないようにする	Memory Bank
		1	ローカル・メモリ (ROM) をシステム側からアクセスできるようにする	
	R	0	ローカル・メモリ (ROM) をシステム側からアクセスできない	
		1 [☆]	ローカル・メモリ (ROM) をシステム側からアクセスできる	
ROM ₀	R/W	0 [☆]	ROM バンクの設定：下	
		1	ROM バンクの設定：上	
WND _{4~0}	R	—	ホスト CPU から見えるローカル・メモリ・アドレスを切り替えるスイッチのビットが読める。内容はスイッチ設定の項を参照	Memory Window
HST _{1,0}	R	—	本体タイプに合わせて設定したスイッチのビットが読める。内容はスイッチ設定の項を参照	
ID _{2,1,0}	R	—	SCSI バス上での SCSI インターフェース・ボードの ID 番号を設定するスイッチ・ビットが読める。内容はスイッチ設定の項を参照	Reset /INT
ILV _{2,1,0}	R	—	システム側に上げる割り込みのレベルを設定するためのスイッチが読み込める。内容はスイッチ設定の項を参照	
RRST	R	0 [☆]	最後に本コマンドを読んだから、他 SCSI 装置からバス・リセットが行われていないことを示す	
		1	SCSI バスの上の $\overline{\text{RST}}$ 信号が 25 μs 以上 “L” アクティブになったとき “1” になる。本コマンド読み込み後、本ビットは “0” になる。ただし、I/O コマンドにより自分自身でバス・ラインをリセットしても本ビットは “1” にはならない 注：SCSI インターフェースでは、パワーオンまたはシステム・リセット時には本ビットを “0” にしているが、同時に SCSI バス上の他の装置からリセットがかかっている可能性があるために、リセット後は必ずしも本ビットが “0” であるとはいえない	

☆はシステムリセット後の設定

書き込みます。ここで指定されたレジスタが CG2H で読み書きできます。

► CG0H : 33C93 の補助ステータス/リード

D₀ : DBR……有効なデータがある

D₁ : PE……パリティ・エラー

D₄ : CIP

D₅ : BSY

D₆ : LCI

D₇ : INT……割り込み確認

► CG2H : 33C93 のコントロール・レジスタ・リード/ライト

図 2-98 は 33C93 のレジスタの一覧です。PC9801-55 ボードでは、これ以外にも 30H~33H が拡張されていて、SCSI ドライブに RESET 信号を送ったりできます。

図 2-99 は 33C93 のコマンド一覧です。コマンド・レジスタで使います。

〈図 2-99〉 WD33C93 のコマンド

コマンド・コード(16進)	コマンド
00	Reset
01	Abort
02	Assert ATN
03	Negate ATN
04	Disconnect
05	Reselect
06	Select With ATN
07	Select Without ATN
08	Select With ATN and Transfer
09	Select Without ATN and Transfer
0A	Reselect and Receive Data
0B	Reselect and Send Data
0C	Wait For Select and Receive
10	Receive Command
11	Receive Data
12	Receive Message Out
13	Receive Unspecified Info Out
14	Send Status
15	Send Data
16	Send Message In
17	Send Unspecified Info In
18	Translate Address
20	Transfer Info
21	Transfer Pad

◆ IDE インターフェース

▶ I/O ポート・アドレス

640H, 642H, 644H, 646H, 648H, 64AH,
64CH, 64EH, 74CH, 74EH

▶ 割り込みレベル

INT₃(IR₉)

▶ DMA チャンネル

なし

IDE インターフェースは、98NOTE や PC9821 以降のデスクトップで使用されています。ソフトウェア的には、SASI インターフェースと同様になっていて、SASI インターフェースと同時に使用できません。

IDE は本来、AT 互換機専用のドライブで、PC/AT の標準 HD インターフェースであった WD1003 のコントロール回路を HDD 内部に搭載したものです。IDE ドライブにアドレス・デコーダとバス・バッファを付けるだけで、バスに直結できるように設計されています。

WD1003 は ST506 規格の HDD 用コントローラで、2 台までの HDD を制御できます。PC98 シリーズ用の SASI インターフェースも、ST506 の HDD に、HD コントローラを付ける形で 2 台まで増設できますから、よく似た形になります。

〈図 2-100〉 IDE の I/O アドレス

レジスタ	I/O ポート・アドレス	R/W	備考
データ・レジスタ	640-641	R/W	16 ビット
エラー・レジスタ	642	R	
ライト・プリコンベンション・レジスタ	642	W	
セクタ・カウント・レジスタ	644	R/W	
セクタ・ナンバ・レジスタ	646	R/W	
シリンダ・ロウ・レジスタ	648	R/W	
シリンダ・ハイ・レジスタ	64A	R/W	
ドライブ/ヘッド・レジスタ	64C	R/W	
ステータス・レジスタ	64E	R	
コマンド・レジスタ	64E	W	
オルタネート・ステータス・レジスタ	74C	R	
ディジタル・アウトプット・レジスタ	74C	W	
ディジタル・インプット・レジスタ	74E	W	

PC98 シリーズでの IDE インターフェースは、BIOS 未対応や増設オプションがない理由で、1 台しか接続できない機種や、使用できる HDD の容量の上限が固定されてしまっている機種もあるようです。PC9801BA2/BX2/PC9821Ap2/As2 等では、500MB の容量を持つドライブや、2 台までの接続を正規にサポートしています。

IDE インターフェースは、16 ビット幅のデータ・バスでデータ転送しますので、内蔵 DMA を使用せずに CPU がソフトウェアでデータ転送します。そのため、高速な CPU を持つ機種では内蔵 DMA よりも高速に転送ができます。

◆ IDE レジスタ

IDE の I/O アドレス・ポートは図 2-100 に示すように、16 ビット・バス×1, 8 ビット・バス×9 の 10 個あります。データの転送は 16 ビット・バスで行われます。

▶ 640H(R/W) データ・レジスタ

データ・レジスタは 16 ビット幅です。コマンドにより次の 5 種類のデータ転送を行います。

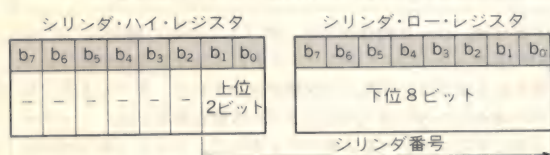
- (1) READ/WRITE…セクタ単位の転送
- (2) READ/WRITE LONG…ECC のデータ転送
- (3) READ/WRITE BUFFER…ドライブ内のデータ・バッファとの転送
- (4) FORMAT…フォーマット用インタリーブ・テープルの転送
- (5) IDENTIFY…ID 情報等のデータ転送

▶ 642H(R) エラー・レジスタ

〈図 2-101〉 エラー・レジスタの各ビット

ビット	名 称	意 味
7	Bad Block Detected	アクセス・セクタ ID に Bad Block マークが検出された (出荷時には Bad Block マークの書き込まれたセクタは存在しない。Format コマンドにより書き込まれる)
6	Data ECC Error	リード・コマンド実行時、データ領域に ECC により回復不能なエラーが発生した
5	—	(未使用)
4	ID Not Found	アクセス要求されたセクタが発見できない。リトライ・モードがイネーブルされている場合は、所定のリトライ動作を行う
3	—	(未使用)
2	Aborted Command	コマンドが実行途中で中断された。原因は、ステータス・レジスタに示される (Not Ready, Write Fault など)。無効コマンドの場合もセットされる
1	Trak0 Error	Restore コマンド実行時にトラック・ゼロが検出されなかった
0	DAM Error	リード・コマンド実行時、データ部のアドレス・マークが検出されなかった

〈図 2-104〉 シリンダ番号のフォーマット



ステータス・レジスタの ERROR がセットされているときは、最後に実行されたコマンドのエラー情報を保持します。パワーオン直後の自己診断や、DIAG NOSE コマンド実行時の結果報告にも使用されます (図 2-101, 図 2-102)。

▶ 642H (W) ライト・プリコンペーション・レジスタ

設定値と機能を図 2-103 に示します。

▶ 644H (R/W) セクタ・カウント・レジスタ

3 種類のコマンドの実行時にパラメータ/返り値の設定用です。

- (1) READ/WRITE/READ VERIFY…転送されるセクタ数
- (2) SET PARAMETERS…1トラック当たりのセクタ数の指定
- (3) POWER CONTROL…動作モードに対応する返り値が設定される

▶ 646H (R/W) セクタ・ナンバ・レジスタ

ディスク・アクセス・コマンド (READ/WRITE/READ VERIFY) 実行時に、アクセスする先頭セクタ番号を指定します。

▶ 648H (R/W) シリンダ・ロウ・レジスタ

アクセスする先頭シリンダ番号の下位ビットを指定

〈図 2-102〉 診断時のエラー・レジスタの値

値	意 味
01	ノー・エラー
02	コントローラ・レジスタ・エラー
03	バッファ RAM エラー
04	ECC 回路エラー
05	CPU ROM/RAM エラー
06・7F	(予約)

〈図 2-103〉 ライト・プリコンペーション・レジスタの値

値	意 味
44H	Read/write Long コマンド時の ECC 長 7 バイト
55H	Read Ahead Cache オフ
AAH	Read Ahead Cache オン
BBH	Read/write Long コマンド時の ECC 長 4 バイト
その他	無効 (Aborted Command Error 発生)

〈図 2-105〉 ドライブ/ヘッド・レジスタ

ビット	名 称	意 味
7	—	ドライブ予約 (1 に設定すること)
6	—	ドライブ予約 (0 に設定すること)
5	—	ドライブ予約 (1 に設定すること)
4	Drive Select	マスタ/スレーブ・モード時 0: マスタ・ドライブ 1: スレーブ・ドライブ シングル・モード時 0: ドライブ 1: ドライブは選択されず、レジスタに 00h が返送される
3~0	Head Select	アクセスの先頭ヘッド番号を設定する。ヘッド番号は 0 から始まる

します。

シリンダ番号のフォーマットを図 2-104 に示します。

▶ 64AH (R/W) シリンダ・ハイ・レジスタ

アクセスする先頭シリンダ番号の上位ビットを指定します。

▶ 64CH (R/W) ドライブ/ヘッド・レジスタ

アクセスするドライブ番号、ヘッド番号を指定します (図 2-105)。

▶ 64EH (R) ステータス・レジスタ・レジスタ

このレジスタを読むとインタラプトがクリアされます (図 2-106)

▶ 64EH (W) コマンド・レジスタ

ドライブに実行させるコマンドを書き込みます。コマンドは 13 種類ですが、ステータス・レジスタのビットが以下の状態のときのみコマンドを受け付けます。

〈図 2-106〉 ステータス・レジスタ

ビット	名 称	意 味
D ₇	BUSY	次の場合に“1”にセットされる。 (1)ホスト・システムから <u>RESET</u> 信号によりハード・リセットされた場合、または Fixed Disk レジスタの RESET ビットによりソフトウェア・リセットされた場合 (2)ホスト・システムからコマンドを書き込まれてからコマンド処理が終了するまで、ただし、データ転送要求中(Data Request ビットが“1”にセットされる)はリセットされる
D ₆	READY	本ビットが“1”で、かつ SEEK COMPLETE ビットが“1”のとき(Seek コマンド直後のみ例外)、ドライブはレディ状態にあり、ホスト・システムからのコマンドを受け付ける。“0”のとき、ドライブはレディ状態(Ready)になく、ホスト・システムからのコマンドを受け付けない。コマンド実行中にノット・レディ(Not Ready)状態が発生した場合は、コマンドは中断され、次のコマンド書き込みまではドライブ状態(Ready/Not Ready)にかかわらず、リセット状態が保持される。このとき、ERROR ビット(ビット 0)もセットされる パワーオン直後はリセットされ、ドライブが定常回転になり、コマンド受信可能状態になった時点でセットされる。 スタンバイ状態で、スタンバイ解除コマンドを受信したときもリセットされ、コマンド受信可能状態になった時点でセットされる
D ₅	WRITE FAULT	“1”はデータ書き込みコマンド実行時、書き込み異常が発生し、データが正しく書き込まれなかったことを示す。このときに ERROR ビット(ビット 0)もセットされ、コマンドは中断される。次のコマンド発行までは、Write Fault 状態の有無にかかわらず、セット状態が保持される。ホスト・システムから次に発行されたコマンドによりリセットされる
D ₄	SEEK COMPLETE	シーク動作をともなうコマンド実行時、シーク動作が正常に終了しなかった場合に“0”にリセットされる。このとき ERROR ビット(ビット 0)もセットされ、コマンドは中断される。つぎのステータス・レジスタ読み取りまでは、リセット状態が保持される。ステータス・レジスタ読み取りが生じると、その時点での SEEK COMPLETE 状態を表示する。シーク・コマンドはシーク動作の終了を待たずにコマンドを正常終了するので、コマンド終了時には本ビットはセットされない。このとき、ドライブはレディ状態にあり、コマンドを受け付ける。パワーオン直後はリセットされ、ドライブが定常回転になり、コマンド受信可能状態になった時点でセットされる。 スタンバイ状態で、スタンバイ解除コマンドを受信したときもリセットされ、コマンド受信可能状態になった時点でセットされる
D ₃	DATA REQUEST	データ転送をともなうコマンド実行時、ドライブがデータ転送準備ができた場合に“1”にセットされる
D ₂	CORRECTED DATA	“1”は、データ・リード時に読み取りエラーが発生したが、ECC により訂正されたことを示す。マルチセクタ・リード動作は中止されない
D ₁	INDEX	ドライブの 1 回転に一度、出力されるパルス信号
D ₀	ERROR	ホスト・システムから発行されたコマンド処理中に、何らかのエラーが発生したことを示す。エラー原因は、本ステータス・レジスタまたはエラー・レジスタに示される。ホスト・システムから次に発行されたコマンドによって、リセットされる(ただし、次に発行されるコマンドが Format コマンド、Write Sectors コマンドの場合、コマンドの書き込み時にはリセットされず、最初のインタラプト発生までの間にリセットされる)。マルチセクタ処理コマンドは中断される

(1) WRITE FAULT=0

(2) READY=1

(3) SEEK COMPLETE=1

このレジスタを読むと、インタラプトがリセットされます(図 2-107)。

▶ 74CH(R) オルタネート・ステータス・レジスタ
「ステータス・レジスタ・レジスタ」と同様ですが、

インタラプトはクリアされません。

▶ 74CH(W) デジタル・アウトプット・レジスタ
デジタル・アウトプット・レジスタ・コマンドは図 2-108 のとおりです。

▶ 74EH(W) デジタル・インプット・レジスタ
デジタル・インプット・レジスタ・コマンドを図 2-109 に示します。

〈図 2-107〉 IDE ドライブ・コマンド

コマンド名	コマンド・コード								使用されるパラメータ・レジスタ					
	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀	DR	CY	HD	SN	SC	WP
Restore	0	0	0	1	X	X	X	X	Y	N	N	N	N	N
Read Sector(s)	0	0	1	0	0	0	L	R	Y	Y	Y	Y	Y	N
Write Sector(s)	0	0	1	1	0	0	L	R	Y	Y	Y	Y	Y	N
Read Verify	0	1	0	0	0	0	0	R	Y	Y	Y	Y	Y	N
Format Track	0	1	0	1	0	0	0	0	Y	Y	Y	N	Y	N
Seek	0	1	1	1	X	X	X	X	Y	Y	Y	Y	N	N
Diagnose	1	0	0	1	0	0	0	0	Y	N	N	N	N	N
Set Parameters	1	0	0	1	0	0	0	1	Y	N	Y	N	Y	N
Power Control	1	1	1	0	0	X	X	X	Y	N	N	N	Y	N
Read Sector Buffer	1	1	1	0	0	1	0	0	Y	N	N	N	N	N
Write Sector Buffer	1	1	1	0	1	0	0	0	Y	N	N	N	N	N
Identify Drive	1	1	1	0	1	1	0	0	Y	N	N	N	N	N
Set Features	1	1	1	0	1	1	1	1	Y	N	N	N	N	Y

レジスタ

DR : ドライブ/ヘッド・レジスタの Drive Select ビット

CY : シリンダ・ハイ/ロウ・レジスタ

HD : ドライブ/ヘッド・レジスタの Head Select ビット

SN : セクタ・ナンバ・レジスタ

SC : セクタ・カウント・レジスタ

WP : ライト・プリコンベンション・レジスタ

Y : コマンド発行に先だちパラメータ設定が必要

N : コマンド発行に先だちパラメータ設定が不要

コマンド・モード

X : Don't care

L : “1” のとき Long コマンド (ECC バイト転送), “0” のときノーマル・リード/ライト

R : “0” のときリトライ実行, “1” のときリトライ禁止

マウス・インターフェース

▶使用 LSI

8255A

▶I/O アドレス

7FD9H, 7FDBH, 7FDDH, 7FDFH (8255)

BFDBH (タイマ)

▶使用割り込み

IR₆ (変更可能な機種もある)

▶初期設定命令

8255=93H

マウスとのインターフェースにはプログラマブル・パラレル・ポートの μ PD8255A の互換品 (以降, 8255 と略す) を使用しています。この **パラレル・ポートを介して, マウス・コントローラ IC を操作** してマウスの移動量と, マウスのトリガ・スイッチが読み出せます。また, マウス用タイマ IC があり, 通常は約 8 ms ごとに割り込みをかけて, マウスの移動量を監視して, マウス・ドライブ内での位置情報やトリガ・スイッチの状態情報を更新し, アイコンでの表示の必要性があれ

ば表示を行います。

タイマからの割り込みは, 8.3 ms~66.7 ms までの 4 種類に変更できるようになっています。割り込み時間の設定は, 8255 とは別アドレスの BFDBH で行います。

使用されている 8255 は, マウス関連情報以外にも, ディップ・スイッチの内容やシステム情報が見えるようになっています。ただし, 機種によって内容が変わります。

一般に 98 用のマウスは 2 ボタン式 (右・左) ですが, EPSON-PC シリーズや, 比較的新しい NEC-98 シリーズ等, 機種によっては本体側は 3 ボタン対応しています。ただ, 98 用 3 ボタン・マウスはあまり市販されてはいないようです。

マウス・インターフェースの 8255 は, ポート A/B/C (下位) がモード 0 で入力に, ポート C (上位) がモード 0 で出力に初期設定されています。モード・セット・コマンドは 93H です。

■ マウスのしくみ

マウスには, X 軸, Y 軸方向の移動に合わせて, パルスが発生するスイッチが付いています。このパル

〈図 2-108〉 デジタル・アウトプット・レジスタ

ビット	名 称	意 味
D _{7~4}	—	(未使用)
D ₃	—	ドライブ予約(1に設定すること)
D ₂	Reset	ホスト・システムによるソフト・リセットとして機能する。“1”の間ドライブはリセット状態となる。このとき内部レジスタではすべてリセットされ、ステータス・レジスタの BUSY ビットがセットされる。マスタ/スレーブ・モード時は、ドライブ/ヘッド・レジスタの Drive Select ビットにかかわらず、両ドライブともリセットされる
D ₁	IEN	“0(アクティブ)” のとき、Drive Select ビット(ドライブ/ヘッド・レジスタ)により選択されたドライブからのホスト・インタラプト信号 IRQ をイネーブルする。 “1” のとき、未処理インタラプト(Pending interrupt)の有無にかかわらず、IRQ 出力はハイ・インピーダンス状態になる
D ₀	—	(未使用)

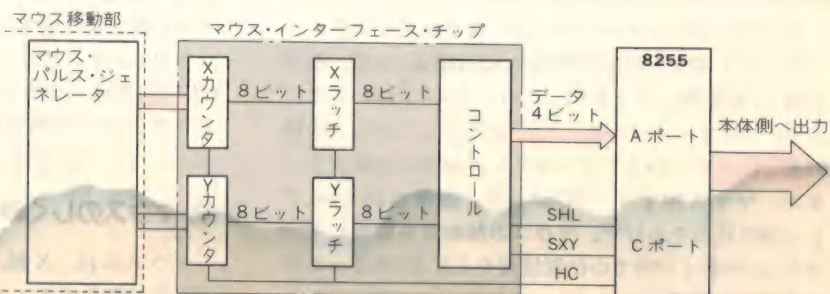
スを、マウス・コントローラ IC の内部の 8 ビットのカウンタで計測して、現在位置を計算します。移動量が 256 カウントを越えてしまうと、マウス・コントローラ IC では、自分の位置を見失ってしまいます。

コンピュータのプログラム(マウス・ドライバ)は、マウスの移動が 256 カウント以内に、再度マウス・コントローラ IC からデータを取得して、その差を算出し絶対位置を計算します。このために、マウス・インターフェースでは、一定時間ごとに割り込み(インタラプト・タイマ)を発生し、最新のデータ取得と計算をします。

マウス・インターフェースの 8255 は、マウス・コントローラ IC のデータを一度に 4 ビットずつしか読み出せません。このため、8 ビットのカウントを、上位 4 ビット/下位 4 ビットに分けて 2 回の操作で読み出し、X/Y 軸で合計 4 回の操作を必要とします。

マウス・インターフェースの構成を図 2-110 に、I/O アドレス一覧を図 2-111 に示します。

〈図 2-110〉
カウンタ IC の内部ブロック図



〈図 2-109〉 デジタル・インプット・レジスタ

ビット	名 称	意 味
D ₇	—	(未使用、読み取り時ハイ・インピーダンス状態となる)
D ₆	—WG	ライト・ゲート信号、ドライブがデータを媒体上に書き込み中にアクティブとなる
D _{5~2}	—Head Select	ドライブ/ヘッド・レジスタの Head Select ビットの 1 の補数を示す
D ₁	—Drive Set 1	ドライブ/ヘッド・レジスタの Drive Select ビットが “1” のとき (Slave Drive 選択), アクティブ (“0”) となる
D ₀	—Drive Set 0	ドライブ/ヘッド・レジスタの Drive Select ビットが “0” のとき (Single モードまたは Master/Slave モードで Master Drive 選択), アクティブ (“0”) となる

◆ ポート A(7FD9H/リード)

▶ D₀~D₃ : MD_{0~3}

マウス位置データで、SXY, SHL で指定されたデータが、バイナリ 4 ビットで出力されます。

▶ D₅ : RIGHT

0 で右トリガ・スイッチが押されています。

▶ D₆ : CENTER

0 で中央トリガ・スイッチが押されています。

▶ D₇ : LEFT

0 で左トリガ・スイッチが押されています。

◆ ポート B(7FDBH/リード)

▶ D₁(D₀~D₁) : SPDSW(FSH, FSL)

CPU クロック・スピードを示します。機種によって異なり使用されていないものもありますが、0 で高速モードです。

▶ D₆ : RAMKL

ディップ・スイッチ SW_{3~6}の設定状態です。0 で内部 RAM の 512KB~640KB を切り離します。

EPSON-PC シリーズでは、ポート B を出力モードにして各種設定を行う機種もありますが、機種ごとに内容が違います。

〈図 2-111〉 マウス・インターフェースの I/O アドレス

L S I	命 令	I/Oポート ・アドレス	R/ W	データ								備 考
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
μ P D 8 2 5 2	ライト・ モード	7FDF	W	1	0	0	1	0	0	1	1	モード・セット
	ライト・ ポート C	7FDF	W	0	0	0	0	1	0	0	0/1	割り込み Enable 0: Enable 1: Disable
		7FDF	W	0	0	0	0	1	1	1	0/1	Clear Count(HC) 0: クリアしない 1: クリアする
	ライト・ ポート C	7FDD	W	HC	SXY	SHL	INT	0	0	0	0	ポート C はこの命令でも変更で きる
	リード・ ポート C (* 1)	7FDD	R	HC	SXY	SHL	INT	MODSM	CPUSW	SW ₁₋₆	SW ₁₋₅	下位 4 ビットによりスイッチの状 態を読み取る
	リード・ ポート B (* 2)	7FDB	R	—	RAMKL	—	—	—	—	SPDSW	—	ディップ・スイッチを読み込む
	リード・ ポート A	7FD9	R	LEFT	×	RIGHT	×	MD ₃	MD ₂	MD ₁	MD ₀	マウスの状態を読み取る
	ライト・ タイマ (* 2)	BFDB	W	0	0	0	0	0	0	T ₁	T ₀	割り込みタイマを設定する

* 1: PC9801/E/F1,2,3/M2,3/U2/VF2/VM0,2,4/UV2 では、下位ビットは未定義

* 2: PC9801RX/RA/DX/DS/DA/CS,PC-98XL/XL2/RL/GS のノーマル・モードではタミー
PC9801/E/F1,2,3/M2,3/U2/VF2/VM0,2,4/UV では存在しない

◆ ポート C 下位(7FDDH/リード)

▶ D₀: SW₁₋₅

ディップ・スイッチ SW₁₋₅の設定状態と RS-232-C
同期モード設定状態を示します。

▶ D₁: SW₁₋₆

ディップ・スイッチ SW₁₋₆の設定状態と RS-232-C
同期モード設定状態を示します。

▶ D₂: CPUSW

ディップ・スイッチ SW₃₋₈の設定状態を示します。
0=80286, 386, 486, Pentium
1=70116, 70136

▶ D₃: MODSW

ノーマル/ハイレゾ・モードの設定状態(ハイレゾ機
種のみ)を示します。

0=ハイレゾ・モード

1=ノーマル・モード

◆ ポート C 上位(7FDDH/ライト)

▶ D₄: INT

0 でマウスのタイマ割り込みを許可します。

▶ D₅: SHL

MD₀₋₃に出力されるデータの切り替えて、0 のとき
下位 4 ビット、1 のときに上位 4 ビットのデータを出力
します(図 2-112)。

▶ D₆: SXY

MD₀₋₃に出力されるデータの切り替えて、0 のとき
X 軸方向、1 のときに Y 軸方向のデータを出力します。

〈図 2-112〉 D₅: SHL

SXY	SHL	データ
0	0	X 軸方向 下位 4 ビット・データ
0	1	X 軸方向 上位 4 ビット・データ
1	0	Y 軸方向 下位 4 ビット・データ
1	1	Y 軸方向 上位 4 ビット・データ

〈図 2-113〉 マウス割り込み周期

値(16 進)	D ₁ D ₀	周波数	時 間
00	0 0	120Hz	8.3ms
01	0 1	60Hz	16.7ms
02	1 0	30Hz	33.3ms
03	1 1	15Hz	66.7ms

▶ D₇: HC

0 のときは、読み出した時点の移動データが MD₀₋₃
に出力されます。

0 → 1 のときに MD₀₋₃のデータがラッチされ、この
際カウンタはクリアされます。

1 のときはラッチされたデータが MD₀₋₃に出力され
ます。

◆ マウス割り込み間隔設定 (BFDBH××・ライト)

マウス割り込み周期は図 2-113 のように D₁~D₀を
設定することで選ぶことができます。

プリンタ・インターフェース

▶使用 LSI

μPD8255A 相当

▶I/O アドレス

40H, 42H, 44H, 46H

▶使用割り込み

IR₈

▶初期設定命令

8255=82H(他にも、システム・ポート用の 8255 のポート C・ビット 7 を使用している)

プリンタ・インターフェースには、μPD8255A の相品(以降、8255 と略す)が使用されています。プリンタへのデータ出力(8 ビット)はポート A が、BUSY 信号入力はポート B・ビット 2 が、ストロブ信号出力はポート C・ビット 7 の合計 10 本の必要最小限の信号線だけになっています。

プリンタ用割り込みとしてポート C・ビット 3 が 8259(スレーブ)の IRQ₀ に接続されていますが、ノーマル・モードでは ACK 信号入力(ポート C・ビット 6)が使用されていないために、8255 をモード 1 に設定できず活用されていません。しかし、ハイレゾ・モードではフル・セントロニクス仕様となっていて割り込みも活用されています。

ノーマル・モードとハイレゾ・モードのプリンタ・インターフェースのハードウェアは異なります。

プリンタ・インターフェース用の 8255 ではプリンタ制御以外にも、ポート B からシステム情報を読み出すことができます。比較的機種依存性の高い情報が多くなっていますので、機種別にどのような信号が出ているか把握したうえで使用しなくてはなりません。

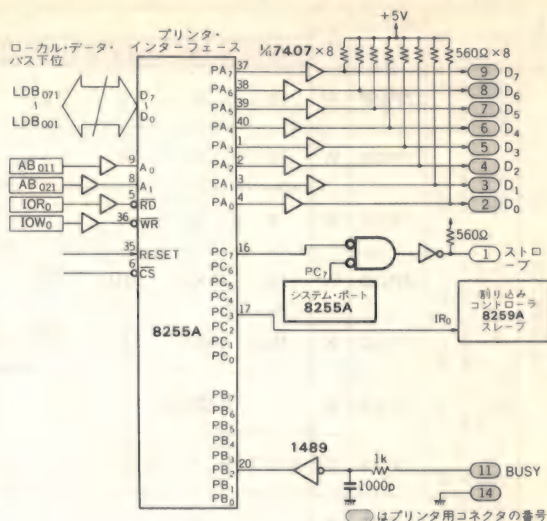
プリンタ・インターフェースの 8255 は、ポート A/C がモード 0 で出力に、ポート B がモード 0 で入力に初期設定されています。モード・セット・コマンドは 82H です。図 2-114 にインターフェース回路を、図 2-115 に I/O アドレス一覧を示します。

◆ ポート A(40H/ライト)

データ出力ポートで書き込んだデータは、プリンタ・インターフェース・コネクタ PDB₀~PDB₇へバッファを通して出力されます。書き込んだデータは 8255 でラッチされるので、前に書いたデータを読み出すこともできます。

8255 とプリンタ・インターフェース・コネクタの間にバッファが入っているために、ポート A で入力も設定しても、プリンタ・ポートを入力として使用することはできません。

〈図 2-114〉 プリンタ・インターフェース回路



◆ ポート B(42H/リード)

▶ D₀ : VF

PC9801VF でのみ“1”になり、その他の機種では“0”になります。

▶ D₁ : CPUT

実際に動作している CPU の種別を示します。EPSON-PC シリーズではディップ・スイッチ SW₃₋₈ の設定がそのまま見えます。
0 : 80286/386/486/Pentium
1 : 70116

▶ D₂ : BSY

プリンタ・コネクタの BUSY にインバータを通してつながっていて、インバータのために論理が反転しています。

0 : BUSY

1 : READY

▶ D₃ : HGC

機能拡張状態を示します。HGC=0 のとき拡張機能使用を示し、16 色表示機能の使用、後続描画機能の使用等を行っているかの状態表示を行います。

EPSON-PC シリーズはディップ・スイッチ SW₁₋₈ の設定が見えます。

▶ D₄ : LCD

プラズマ・ディスプレイ使用/未使用を示します。LCD=0 でプラズマ・ディスプレイ使用です。

EPSON-PC シリーズはディップ・スイッチ SW₁₋₃ の設定が見えます。

▶ D₅ : MOD

CPU クロックの状態を示します。システム・クロッ

〈図 2-115〉 プリンタ I/O アドレス

μPD8255	命 令	I/Oポート・アドレス	R/W	データ								備 考
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
プリンタ・ポート・コントロール・レジスタ	ライト・モード	46	W	1	0	0	0	0	0	1	0	8255A モード・セット
	ライト・シグナル 1	46	W	0	0	0	0	0	0	1	1/0	80287/80387 (SX) のリセット制御 0: シャットダウン時リセットしない 1: リセットする (* 1)
		46	W	0	0	0	0	0	1	1	1/0	IR ₈ の ON/OFF 0: アクティブ 1: インアクティブ
		46	W	0	0	0	0	1	1	1	1/0	PSTB の ON/OFF 0: アクティブ 1: インアクティブ
ポート C	ライト・シグナル 2	44	W	$\overline{\text{PSTB}}$	0	0	0	IR ₈	0	RST287 (387)	0	PSTB, IR ₈ , RST287/387 は本命令でも制御可能 (* 1)
ポート A	ライト・データ	40	W	WD ₈	WD ₇	WD ₆	WD ₅	WD ₄	WD ₃	WD ₂	WD ₁	プリンタにデータを送る
	リード・データ (診断用)	40	R	WD ₈	WD ₇	WD ₆	WD ₅	WD ₄	WD ₃	WD ₂	WD ₁	ライト・データでセットしたデータを読み込む
ポート B	リード・シグナル 1 (* 2)	42	R	TYP ₁	TYP ₀	MOD	LCD	HGC	$\overline{\text{BSY}}$	CPUT	VF	プリンタの状態および CPU のモード・タイプを読み込む
ポート C	リード・シグナル 2	44	R	$\overline{\text{PSTB}}$	×	×	×	IR ₈	×	RST287 (387)	×	8255A のポート C の状態を読み込む
システム・ポート・コントロール・レジスタ	ライト・ポート C (* 3)	37	W	0	0	0	0	1	1	0	1/0	PSTB 信号 Enable F/F の ON/OFF 0: アクティブ 1: インアクティブ

* 1: RST287/387 は 80286/386/486/Pentium 搭載機種のみ

* 2: PC9801/E/F1,2,3/M2,3 では、LCD, HGC, CPUT は未定義

VF は、PC9801VF2 でのみ 1, 他はつねに 0, PC9801 では $\overline{\text{BSY}}$ のみ使用, PC9801 では MOD も未定義

* 3: PC9801 では、 $\overline{\text{PSTB}}$ Enable F/F は、I/O ポート・アドレス 94H (IMB フロッピー・ディスク・インターフェースの外付けレジスタ) の D₁ ビットを使用

×: 不定

IR₈: プリンタ制御回路から 8259 への割り込み信号

クの判別 (5 MHz/8 MHz) ができ、8253 (タイマ LSI) の入力クロックを知るために使用します。

0: 5/10/12/20/25/40 MHz (5/10 MHz 系)

1: 8/16/33/60/66 MHz (8 MHz 系)

CPU クロックとシステム・クロックとの関係は、機種によっては上記の内容と異なる場合がありますが、MOD で得られるデータは初代 PC9801 を除いて、8253 の入力クロックを表します。

初代 PC9801 は、プリンタ・コネクタの 13 ピンに接続されていますので、システム・クロックを知ることはできません。

▶ D₆~D₇: TYP₀, TYP₁

システムのタイプを図 2-116 のように設定します。

〈図 2-116〉 システム・タイプの設定

TYP ₁	TYP ₀	システム・タイプ
0	0	PC9801
1	1	PC9801U2
0	1	未定義
1	0	上記以外の機種

■ ポート C (モード 0・44H/出力)

▶ D₁: RST287 (RST387)

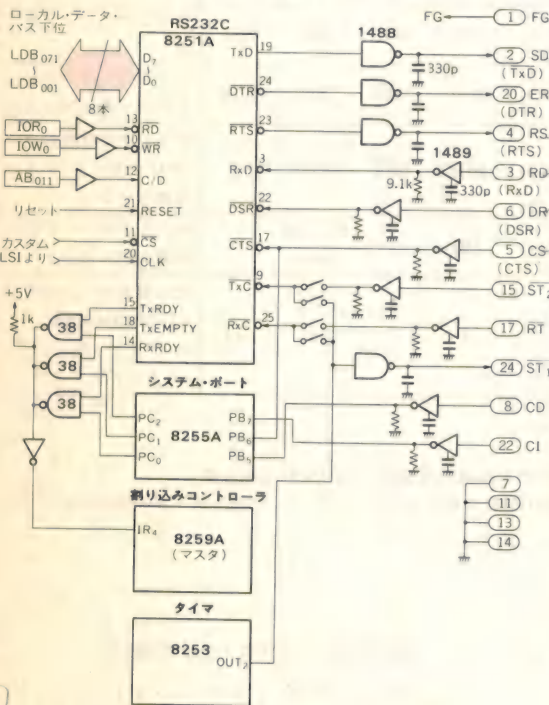
CPU リセット発生時、NDP または CPU 内蔵の NDP 機能をリセットするかどうかの指定です。このビットは 80286/386/486/Pentium CPU 動作時のみ意味を持ちます。

〈図 2-118〉 RS-232-C 関連 I/O アドレス

LSI	命 令	I/Oポート・アドレス	R/W	データ								備 考
				D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
8251	モード(A)	32	W	S ₂	S ₁	EP	PEN	L ₂	L ₁	B ₂	B ₁	μPD8251 動作モードの設定(非同期)
	モード(B)	32	W	SCS	ESD	EP	PEN	L ₂	L ₁	0	0	μPD8251 動作モードの設定(同期)
	コマンド	32	W	EH	IR	RS	RST	SBR	REN	ER	TEN	
	ステータス	32	R	DR	SYN	FE	OE	PE	TE	RRDY	TRDY	
	データ・リード	30	R	RD ₈	RD ₇	RD ₆	RD ₅	RD ₄	RD ₃	RD ₂	RD ₁	
	データ・ライト	30	W	SD ₈	SD ₇	SD ₆	SD ₅	SD ₄	SD ₃	SD ₂	SD ₁	
8253	カウンタ・セット	75	W	C ₇	C ₆	C ₅	C ₄	C ₃	C ₂	C ₁	C ₀	
	カウンタ・モード	77	W	SC ₁	SC ₀	RL ₁	RL ₀	M ₂	M ₁	M ₀	BCD	
8255	マスク・セット	35	W	×	×	×	×	×	TXRE	TXEE	RXRE	
	リード・シグナル	33	R	CI	CS	CD	×	×	×	×	×	CI は PC 9801 では無効

×印：不定

〈図 2-117〉 RS-232-C インターフェース



0：リセットしない

1：リセットする

▶ D₅：IR_s

プリンタ用割り込みです。8259(スレーブ)のIRQ₁に接続されていますが、前述のとおり、ノーマル・モードでは利用されていません。

0：ON

1：OFF

CPU が 80286/386/486/Pentium で動作している場合は、IR_sは NDP が使用します。

〈図 2-119〉 8251 に必要なクロック

動作条件	8251 入力クロック	最大遅延時間 (28 クロック)
システム・クロック 8MHz	1.9968MHz	14.03μs
システム・クロック 5/10MHz	2.4576MHz	11.40μs
PC9821Af, Ne	9.8304MHz	2.85μs

注意：ステータスの更新はステータスに影響を与える事象が起こってから最大 28 クロック(μPD8251 の入力クロック)周期の遅延がある。したがってステータス更新までに必要な最大遅延時間は上のようになる。I/O ポートを参照する場合は注意すること

▶ D₇：PSTB

プリンタ・ストローブ信号の出力です。システム・ポートの PC₆と AND を取って、プリンタ・コネクタの PSTB に接続されています。プリンタ・ポートやシステム・ポートの 8255 を初期化する場合、その設定順序に注意が必要です(詳細はシステム・ポートの章参照)。

0：LOW

1：HIGH

RS-232-C インターフェース

▶使用 LSI

μPD8251A 相当

▶ I/O アドレス

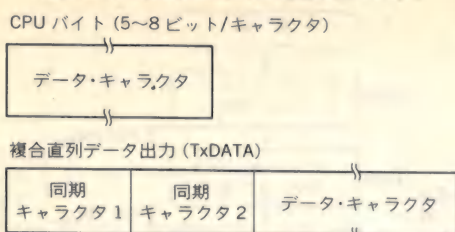
30H, 32H

▶使用割り込み

IR_s(ベクタ# 0CH)

98 シリーズの RS-232-C インターフェースには、

〈図 2-120〉 同期モードでの送信フォーマット



μPD8251A の互換品 (以下, 8251 と略す) を使用しています。その他にも, ボーレート生成用にタイマ LSI (8253) と割り込みマスク, 外部信号読み出し用にパラレル・ポート (8255) を使用しています。

図 2-117 にインターフェース回路を, 図 2-118 に関連 I/O アドレス一覧を示します。

8251 について

8251 は, 同期/調歩同期 (非同期) モードを持ちます。同期モードでは, 同期キャラクタ数が 1~2, 内部/外部同期検出, 自動同期キャラクタ挿入等が可能になります (98 シリーズでは外部同期検出に必要な SYNC 端子が RS-232-C コネクタに接続されていないので, 外部同期検出モードは使えない)。

8251 に必要なクロックには 2 種類あって, TxC/RxC は RS-232-C の通信ボーレートを決定するために, 8253 で発生されたクロックが入力されています。

もう一つの CLK は, LSI 内部の動作タイミングを作るもので, LSI の処理速度に影響を与えますが, ボーレート等とは無関係です。クロックには, 1.9968 MHz か, 2.4576 MHz, 9.8304 MHz が入力されています (図 2-119)。

8251 の仕様では, CLK のクロックは, TxC/RxC のクロックの 4.5 倍 (同期モード時は 30 倍) である必要があります。例えば, CLK に 2.4576 MHz を使用している場合は, 計算上では 34133 bps 以上出せないことになりますので, 実質的には 19200 bps が上限になってしまいます。しかし, 実際には 38400 bps で使用しても問題はなさそうです。

調歩同期モードではボーレート設定を, $\times 1$, $\times 16$, $\times 64$ の 3 種類から選べます。これは, TxC/RxC から入力されたクロックから分周して通信ボーレートを作るときの分周比を指定するもので, 一般的には $\times 16$ を使用します。 $\times 1$ を使用した場合は, 同期通信と同様に TxC/RxC に相手側の 8251 と同一位相のクロックを与える必要があります。

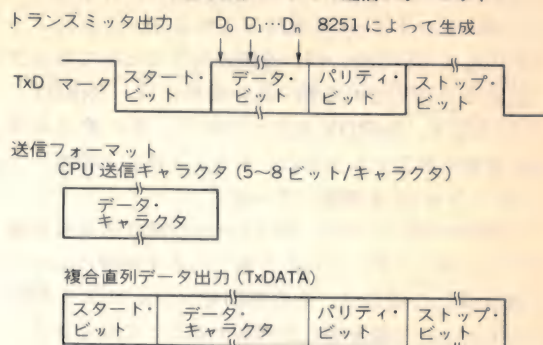
●ボーレート・ファクタの特徴

$\times 1$ 最も高速な通信が可能。

ただし, 送受信で相手と CLK 同期を取る必要がある。

$\times 16$ 同期を取る必要が特になく, 一般的な速度をサ

〈図 2-121〉 調歩同期モードでの送信フォーマット



ポート。

$\times 64$ 同期を取る必要が特になく, 内部分周比が大きいので低速通信向け。

同期モードと調歩同期モード

▶同期モード

送信側が書き込んだデータをそのままシリアルに変換して受信側に送る方式で, 各キャラクタの先頭ビットを検出する (同期を取る) 必要があります。同期を取る方法としては, 以下の二つが挙げられます。一つは特定ビット列 (同期キャラクタ) のサンプリングで同期を取る方法です。

もう一つは, 端子入力によって同期を取る方法があります。

同期モードは, 一般にはあまり使われていないモードで, 専用モデム等に使われます。

同期モードでの送信フォーマットを図 2-120 に示します。

▶調歩同期モード

送信側が書き込んだデータの単位ごと (5~8 ビット) に特定の信号を付加して, 受信側に送ります。その特定の信号には, スタート・ビットとストップ・ビットとパリティ・ビットがあります。スタート・ビットは各キャラクタの先頭ビットの前に, ストップ・ビットは各キャラクタの後尾ビットの後に付加します。パリティ・ビットを付加する場合は, 後尾ビットとストップ・ビットの間に挿入されます。

これらの付加されたビットのために, 同期モードに比べて転送効率が悪くなりますが, 送信側と受信側の双方の同期を取る必要がないので, 手軽に使用でき, 一般的に使われています。汎用の RS-232-C は, ほとんどこの調歩同期方式になっています。

調歩同期モードでの送信フォーマットを図 2-121 に示します。

8255 (システム・ポート)

8251 単体だけでは読み取ることができない外部信

号を読み出すために、システム・ポート用の 8255 のポート B・ビット 5～7 を使用しています。これによって、CD、CTS(CS)、CI、を読み出すことができます。

8251 から出力される割り込み信号には、TxRDY、TxEMPTY、RxRDY の三つがあり、それぞれの有効、無効を設定するためにシステム・ポートのポート C・ビット 0～2 を使用しています。

内蔵 RS-232-C では、IR₄ の一つの割り込みしか使用できませんので、これらの割り込みを同時に二つ以上有効にした場合は、その判別をソフトにより 8251 のステータスを読み出して判別することになります。

◆ 8253(タイマ LSI)とボーレート

RS-232-C インターフェースのボーレートは、タイマ LSI の 8253 で生成されます。使用しているのは #2 チャンネルで、モード 3(方形波レート・ジェネレータ)で使われています。8253 に入力されるクロックは機種によって 2 種類あり、これを認識して 8253 の分周比を決めないとボーレートが合わなくなるので注意が必要です。

8253 に入力されるクロックは、CPU クロックが 8 MHz 系のときは 1.9968 MHz、5 MHz 系のときには 2.4576 MHz になります。これを調べるには、プリンタ・ポート用の 8255 の 42H のビット 5 を読み取り、1 ならば 1.9968 MHz、0 ならば 2.4576 MHz であることがわかります。

RS-232-C では、普通、1200、2400、4800……等と倍々のボーレートを使用します。8251 で調歩同期式(非同期)を使う場合は、ボーレートの 16 倍のクロックが必要になるので、19200 bps では、307.2 kHz が必要な計算になり、5 MHz 系の場合は 1/8 に分周すれば良いのですが、8 MHz 系の場合は 1/6.5 となり、8253 では分周できません。この理由から、8 MHz 系クロックを持つ機種では RS-232-C のボーレートの上限が 9600 bps に限定されてしまいます。

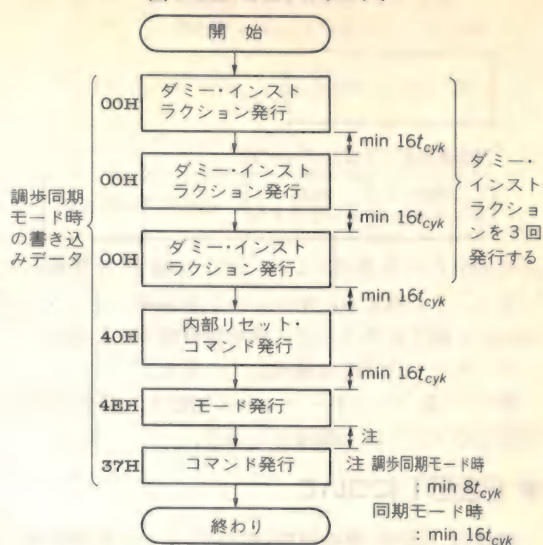
◆ 8251 のレジスタ

8251 の I/O ポートは二つあり、「モード・コマンド書き込み(設定)/ステータス読み出しポート」と、「データの読み書きポート」です。

モード・コマンド設定ポートは一つのアドレスを共用しており、モード設定(初期化データ 5EH を書き込む)は 8251 をリセットした直後に一度だけ設定でき、二度目からはコマンド設定ポートになります。8251 のリセットはコマンド設定(ソフトウェア)から行うので、図 2-122 の手順でリセット→モード設定する必要があります。

8251 の各レジスタの内容は以下のとおりです。

〈図 2-122〉 8251 のリセット



◆ モード設定(32H/ライト)

モード設定には、同期モード、調歩同期モードの 2 種類があり、D₆～D₇ のビットの意味が変わります(図 2-123)。

► D₆～D₁ : B₀～B₁

送受信のボーレートと、TxCLK、RxCLK の関係を規定します。ボーレートに対して送受信クロックの周波数が 1 倍か、16 倍か、64 倍かを選択します。

同期モードでは、B₀=0、B₁=0 にします。

・ D₂～D₃ : L₀～L₁

1 キャラクタのビット数の設定に用います。このビット数にはパリティ・ビット等の付加ビットは入りません。

プログラムしたキャラクタ長が 8 ビットより少ない場合は、上位桁のデータが無効になります。無効になったビットは、読み出し時は「0」になり、書き込み時は無視されます。

► D₄～D₅ : P₀～P₁

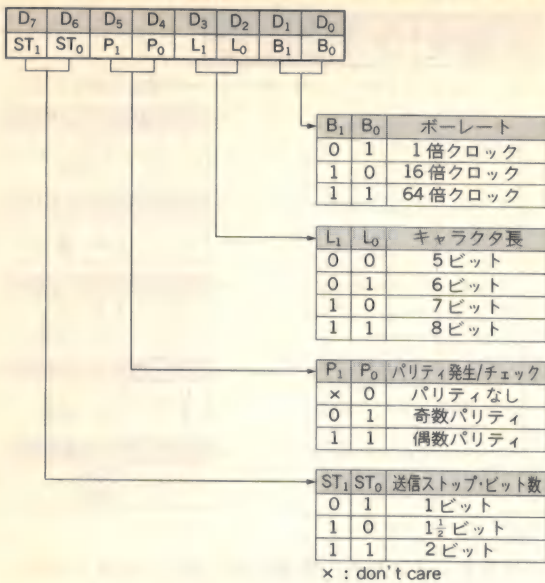
パリティ・ビットの発生(送信)やチェック(受信)機能を制御します。パリティ発生/チェックは、キャラクタ・ビットとパリティ・ビットを合わせたビットの中で「1」であるビットの数が偶数(偶数パリティ)、または奇数(奇数パリティ)になるように、パリティ・ビットを発生/チェックします。

◆ 調歩同期モード特有のビット

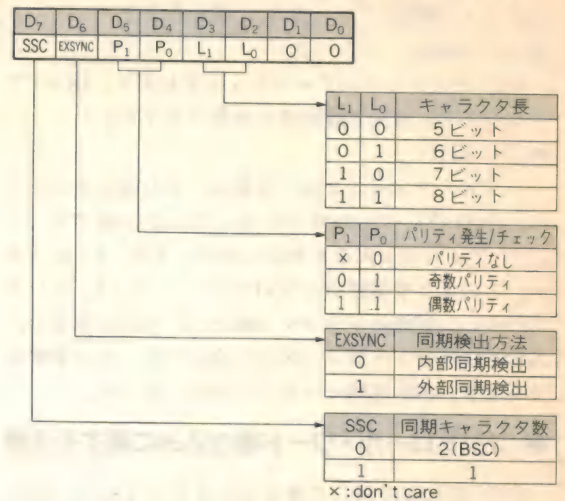
► D₆～D₇ : ST₀～ST₁

送信時に付加するストップ・ビットの長さの指定に用います。受信動作には影響を与えません(データ受信時には 1 ビットのストップ・ビットのみチェックされる)。

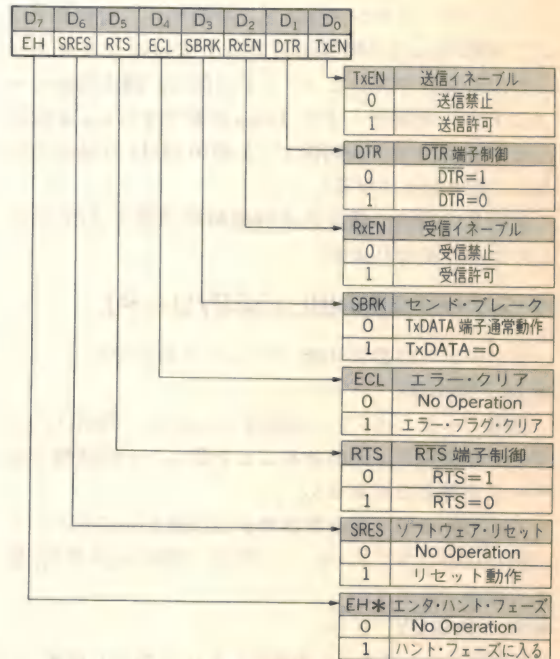
〈図 2-123 (a)〉 調歩同期モードにするためのモード・ワード



〈図 2-123 (b)〉 同期モードにするためのモード・ワード



〈図 2-124〉 コマンド・ワード



* : EH (D₇) ビットは同期モードでのみ有効で、調歩同期モードでは Don't care となる。

◆ 同期モード特有のビット

▶ D₆ : EXSYNC

同期検出方法を選択します。外部同期検出にプログラムした場合には、キャラクタ同期のための同期キャラクタの受信は必要ありません。

▶ D₇ : SSC

同期キャラクタ数を決めます。モード・ワードの次に SSC ビットによって設定された数の同期キャラクタを書き込みます。

◆ コマンド設定 (32H/ライト)

コマンド設定のフォーマットを図 2-124 に示します。

▶ D₀ : TxEN

送信の許可/禁止を指示します。送信禁止 (TxEN = 0) にすると、その時点で書き込まれているデータをすべて送出してから送信を停止します。

▶ D₁ : DTR

8251 の汎用出力ポートの制御用です。「DTR=1」ならば RS-232-C コネクタの DTR 端子は -V になり、「DTR=0」ならば +V になります。

▶ D₂ : RxEN

受信の許可/禁止を指示します。RxEN=0 で受信禁止です。同期モードの場合、受信禁止を行うと比同期状態になります。

▶ D₃ : SBRK

ブレイク信号の送出用で、SBRK=1 のとき現在送出中のデータを無効にして、8251 の TxD 出力を L レベルにします。SBRK=0 でブレイク状態が解除さ

れます。なお、この機能は送信禁止状態でも有効です。

▶ D₄ : ECL

8251 のエラー・ステータス (PE, OE, FE) のクリアを行います。ER=1 でエラーで、クリア「0」されます。ハント・フェーズに入るとき (EH=1) や受信許可 (RxEN=1) にするときは、同時に「ECL=1」にします。

▶ D₅ : RTS

8251 の汎用出力ポートの制御用です。「RTS=1」ならば RS-232-C コネクタの DTR 端子は -V になり、

「RTS=0」ならば+Vになります。一般的には、受信データの抑制(フロー操作)に使われます。

▶ D₆: SRES

8251 をソフトウェア・リセットさせます。IR=1 でリセットし、モード設定待ち状態になります。

▶ D₇: EH

ハント・フェーズとは、同期モードを確立するために、RxDのレベルの変化を待っている状態です。ハント・フェーズに入るときはこのビットを「1」にします。このとき受信許可のRxEN ビットも「1」にしてください。同期キャラクタを検出し、同期を取ると、自動的にハント・フェーズから抜けてデータの受信が始ります。調歩同期モードでは使用しません。

◆ コントロール・ワード書き込みに関する注意

CPU が LSI に対して書き込みを行ってから、次の書き込みを行うまで、十分なタイミングを取る必要があります(書き込み回復時間)。

リセット・コマンド実行と、次のモード指定までには、最低 $6t_{cyk}$ 分の時間をあけなくてはなりません。それ以外のコマンドとコマンドの間は、調歩同期モードで $8t_{cyk}$ 、同期モードで $16t_{cyk}$ 必要です(t_{cyk} は 8251 の CLK, 1 サイクル時間で、2.4576 MHz の場合では $1t_{cyk}$ = 約 $0.4\mu s$ になる)。

初期化の場合の書き込み回復時間は図 2-122 に示したとおりになります。

◆ ステータス読み出し(32H/リード)

ステータスは図 2-125 に示したとおりです。

▶ D₀: TxRDY

送信データ・バッファ状態を示します。TxRDY=0 でバッファにデータがあることを示し、この状態ではデータを送出できません。

TxRDY の割り込み要求端子の状態と同じです。この割り込みはマスク・セット(35H)で割り込み許可/禁止を設定できます。

▶ D₁: RxRDY

RxRDY=1 でデータを受信したことを示します。

RxRDY の割り込み要求端子の状態と同じです。この割り込みはマスク・セット(35H)で割り込み許可/禁止を設定できます。

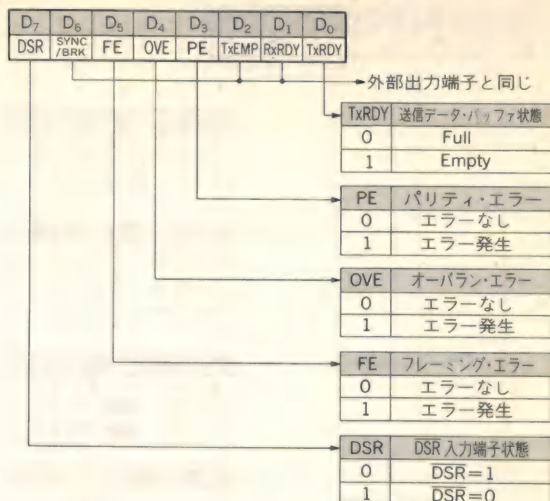
▶ D₂: TxEMP

送信データ・バッファ(第2バッファ)とトランスミッタ内の送信バッファ(第1バッファ)が共に空であることを示します。

TxEMP の割り込み要求端子の状態と同じです。この割り込みはマスク・セット(35H)で割り込み許可/禁止を設定できます。

▶ D₃: PE

〈図 2-125〉 ステータス



パリティ・エラーの発生を示します。エラーがあれば「1」、なければ「0」になります。エラーが発生しても 8251 の動作は停止しません。ER=1 でエラーはクリアされます。

▶ D₄: OVE

オーバラン・エラーの発生を示します。CPU が受信データの読み出しに遅れたときに「1」になります。エラーが発生しても 8251 の動作は停止しません。ER=1 でエラーはクリアされます。

▶ D₅: FE

フレーミング・エラーの発生を示します。ストップ・ビットが検出されなかったときに「1」になります。エラーが発生しても 8251 の動作は停止しません。ER=1 でエラーはクリアされます。

▶ D₆: SYNC/BRK

調歩同期モードでは、ブレイク信号(RxD が 2 キャラクタ以上の時間「0」になった場合)を受信したときに「1」になります。同期モード・内部同期検出の場合は、同期キャラクタを検出したときに「1」になります。

▶ D₇: DSR

汎用入力ポートの DSR の状態を示します。RS-232 -C コネクタの DTR 端子が -V のときに「DSR=1」になり、+V ならば「DSR=0」になります。

◆ リード・シグナル(33H/リード)

CD, CS(CTS), CI の読み出しには、システム・ポートのポート B を使用しています。

• D₅: CD RS-232-C

• D₆: CS(CTS) RS-232-C

• D₇: CI RS-232-C

RS-232-C 用に使用されている 8251 では読み取るこのことのできない、CI, (CTS)CS, CD の信号を読み

〈図 2-126〉 拡張 RS-232-C の I/O アドレス

L S I	命 令	I/O ポート ・ アドレス		R/ W	データ								備 考
		CH ₂	CH ₃		D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
μ P D 8 2 5 1	モード(A)	B3	BB	W	S ₂	S ₁	EP	PEN	L ₂	L ₁	×	×	μPD8251 動作モードの設定(非同期)
	モード(B)	B3	BB	W	SCS	ESD	EP	PEN	L ₂	L ₁	0	0	μPD8251 動作モードの設定(同期)
	コマンド	B3	BB	W	EH	IR	RS	RST	SBR	REN	ER	TEN	
	ステータス	B3	BB	R	DR	SYN	FE	OE	PE	TE	RRDY	TRDY	
	データ・リード	B1	B9	R	RD ₈	RD ₇	RD ₆	RD ₅	RD ₄	RD ₃	RD ₂	RD ₁	
	データ・ライト	B1	B9	W	SD ₈	SD ₇	SD ₆	SD ₅	SD ₄	SD ₃	SD ₂	SD ₁	
	マスク・セット	B0	B2	W	×	×	×	×	×	TXR	TXE	RXR	
	リード・シグナル	B0	B2	R	$\overline{\text{CI}}$	$\overline{\text{CS}}$	$\overline{\text{CD}}$	×	×	×	×	×	
	割り込みレベル・センス	B0	B2	R	×	×	×	×	×	×	IR ₁	IR ₂	

取ることができます。RS-232-C の各信号が、+V (OFF) のときや開放されているときに、対応するビットが“1”になります。-V (ON) のときは“0”です。

◆ マスク・セット(35H/ライト)

RS-232-C の割り込みマスク許可/禁止には、シテム・ポートのポート C を使用しています。

- ・ D₀ : RxRDY 割り込みイネーブル
- ・ D₁ : TxEMPTY 割り込みイネーブル
- ・ D₂ : TxRDY 割り込みイネーブル

RS-232-C 用割り込みのマスク用です。“1”で割り込み禁止，“0”で割り込み可能になります。

拡張 RS-232-C インターフェース

▶ 使用 LSI

μPD8251A 相当×2

▶ I/O アドレス

B1H, B3H (チャネル#2)

B9H, BBH (チャネル#3)

BOH, B2H (割り込みセンス, リード・シグナル)

▶ 使用割り込み

INT₀₋₆ (任意に設定できる)

オプションで拡張スロットに RS-232-C インターフェースを増設できます。NEC 純正として PC9861/K というオプションがありますが、いくつかのサード・パーティからも似たようなボードが販売されています。

使用 LSI に、μPD8251A 相当を二つ使い、内蔵 RS-232-C インターフェースと、ほぼ同様の手順で操作できます。違いは、ボーレート・ジェネレータを拡張ボード上に持ちハードウェア(ディップ・スイッチ)で変更することと、割り込みレベルをハードウェア(ディップ・スイッチ)で変更できることです。

〈図 2-127〉
割り込みレベル

IR ₁	IR ₂	INT レベル	
		CH ₂	CH ₃
0	0	INT ₀	INT ₀
0	1	INT ₁	INT ₄
1	0	INT ₂	INT ₅
1	1	INT ₃	INT ₆

AIWA の B98-01 では、××D1H, ××D3H, ××D5H, ××D7H(××は任意に設定できる)で、ボーレートの変更や、自己診断機能の ON-OFF が設定できます。拡張 RS-232-C の I/O アドレス一覧を図 2-126 に示します。

◆ リード・シグナル/割り込みレベル・センス (CH₂: B0H/CH₃: B2H)

▶ D₀~D₁: IR₁~IR₂

ディップ・スイッチで設定した割り込みレベルを読み出せます。B98-01 では「D₂(IR₃)」も使用されています。割り込みレベルを図 2-127 に示します。

- ・ D₅ : CD RS-232-C
- ・ D₆ : CS (CTS) RS-232-C
- ・ D₇ : CI RS-232-C

RS-232-C 用に使用されている 8251 では読み取ることのできない、CI, (CTS)CS, CD の信号を読み取ることができます。RS-232-C の各信号が、+V (OFF) のときや開放されているときに、対応するビットが“1”になります。-V (ON) のときは“0”です。

◆ マスク・セット(CH₂: B0H/CH₃: B2H)

- ・ D₀ : RxRDY 割り込みイネーブル
- ・ D₁ : TxEMPTY 割り込みイネーブル
- ・ D₂ : TxRDY 割り込みイネーブル

RS-232-C 用割り込みのマスク用です。“1”で割り込み禁止，“0”で割り込み可能になります。

3

拡張スロットの信号と使い方



PC 98 シリーズの拡張スロットの詳細

◆ 拡張スロットの種類

PC98 シリーズの拡張バスは、大きく分けると、デスクトップ系、ノート系、NESA バス等があります。そのうち最も一般的なものは、初代 PC9801 からの流れをくむデスクトップ系の拡張スロットで、拡張基板の外形や、寸法は初代から現在まで変わりません。

拡張スロットの信号線(98 バス)は全部で 50 本あり、データ・バス 16 ビット、アドレス・バス 24 ビット(8086/V30 系は 20 ビット)等が出ています。

NEC のノート系は、拡張基板が挿入できるスロットではなく、拡張バス・コネクタが装備されています。このコネクタは、従来のデスクトップ系とほぼ同じ信号線に加えて、増設用 FDD や CRT 関連の信号が追加され、合計 110 本あります。専用の I/O 拡張ユニットを使用することで、デスクトップ用の拡張基板を使用することもできます。

EPSON のラップトップ、BOOK 系の機種には、L スロットと呼ばれる拡張スロットが搭載されています。デスクトップ系のスロットとは形状や寸法が小さくなっていますが、信号線は 50 本で、98 バスとほぼ同等な規格になっています。

EPSON の NOTE 系の機種のバスは、EPSON 独自規格の 80 本バスと、NEC ノートと同一の 110 本バスの 2 種類があります。PC286NOTEExecutive から、PC386NOTE WR までは 80 本バスで、PC386NOTE AE 以降には 110 本バスが搭載されています。110 本バスは NEC のノートのバスとの互換性があります。

80 本のバスと 110 本バスの違いは、増設 FDD の信号がない、アドレス・バスの $A_{19} \sim A_{23}$ がない、CRT 出力信号の違い等があります。80 本バスを持つ機種には増設 FDD コネクタが別に付いています。

NESA バスは、PC-H98 シリーズに搭載されているバスで、バス幅を従来の 16 ビットから 32 ビットに拡張し、新しいアーキテクチャによる高性能なバス

です。PC-H98 シリーズにしか搭載されず、使用ユーザが増えないためか、拡張基板の価格が割高なようです。

◆ 拡張スロットの数

拡張スロットの数は機種によって違います。初代 PC9801 や PC9801E では、FDD/HDD インターフェースが標準搭載されていなかったためもあって 5~6 個付いていましたが、一般的なデスクトップは 4 個が標準になっています。デスクトップでもローコスト版の機種では 3 個に減らされているものもありますし、小型化された機種では 2 個、PC286C(EPSON)では 1 個しかありません。

◆ 拡張基板の形状と寸法

拡張基板の外形、寸法は図 3-1 のようになっています。バス・コネクタは、カード・エッジ・コネクタで、基板にはカード・エッジ・パターンが 2.54 mm ピッチで 50 個×両面で合計 100 個あります。

拡張基板の端子は、はんだ面(裏面)が A、部品面(表面)が B で、 $A_{01} \sim A_{50}$ と $B_{01} \sim B_{50}$ となっています。

スロット一つの高さは 25 mm ありますが、基板上面からのスペースは 20 mm ほどしかありません。また、基板の抜き差し用にカード・プラが取り付けられます。

参考に EPSON・L スロット基板の外形と寸法を図 3-2 に示します。

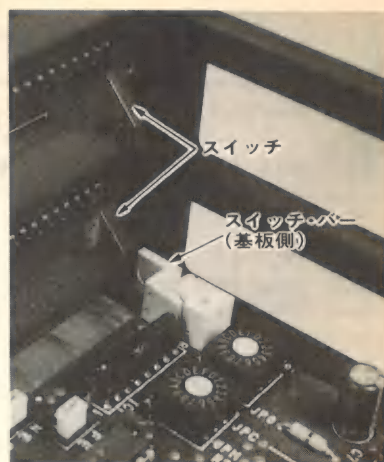
◆ 機種による拡張バスの違い

PC98 シリーズのアドレス・バスは 2 種類あります。8086/V30 等の 1M バイトしかアドレス空間(20 ビット)を持たない CPU 用バスと、80286 以降の 16M バイト以上のアドレス空間(24 ビット)を持つ CPU 用バスです。

拡張スロットには、8086/V30 用の機種にも 24 ビット分のアドレス端子が用意されていますが、使用されてはいません。そのために、これらの機種用に設計された拡張基板では、上位 4 ビット分のアドレス(A_{20}

Technical drawing of a rectangular component with dimensions and labels:

- Top edge: 1.6 ± 0.2
- Top-left corner: $2-R1.0$, $2-C1.0$
- Top-right corner: 7.62
- Right edge: 5.08 , 20.32 , 143.7 ± 0.3
- Bottom edge: 5.0 ± 0.1
- Bottom-left corner: 9.1 ± 0.2 , 2.32 ± 0.05
- Left edge: 148.7 ± 0.4 , 129.1 ± 0.1 , 124.46 ± 0.15
- Internal dimensions: $2.54 \times 49 = 124.46 \pm 0.15$
- Internal width: 10.0 ± 0.2
- Internal length: 164.4 ± 0.3 , 169.4 ± 0.4
- Labels:
 - 部品実装禁止 (No component assembly)
 - 表面端子 B₅₀ (Surface terminal B₅₀)
 - 裏面端子 A₅₀ (Back terminal A₅₀)
 - 表面端子 B₁ (Surface terminal B₁)
 - 裏面端子 A₁ (Back terminal A₁)
 - カード・ブラ取り付け孔 (Card/Bra mounting hole)
- Inset drawing (bottom left):
 - Dimensions: 1.8 ± 0.1 , 7.62 ± 0.2 , 12.54 ± 0.05 , $R0.9$
 - Feature: 2- $\phi 40 \pm 0.15$ (2 holes of diameter 40 ± 0.15)



86
10
66
5
5
54
54
2-φ3.5
A 部
□ 10 GND
部品面高さ制限
上スロット 8.5mm
下スロット 12.5mm
(コネクタ出力部は上スロットのはんだ面寸法との関係をとれば 13.8mm まで可)
はんだ面高さ制限
上スロット 2.0mm
(下スロットに WT4-1R, WT4-M12R BOARD がきた場合 A 部は 1.0mm)
下スロット 2.5mm
部品禁止
ジレリスト逃げる
124.7
4
62.23 ± 0.1
B50
A50
(はんだ面)
64.57 ± 0.1
B1
A1
(はんだ面)
38 ± 0.1
9

〈図 3-3 (a)〉 バス・スロット信号一覧(デスクトップ・タイプ)

● A 面

● B 面

8086/V30 タイプ・バス				80286/80386/80486/ Pentium タイプ・バス			8086/V30 タイプ・バス				80286/80386/80486/ Pentium タイプ・バス		
端子 番号	信号名	方向	機 能	信号名	方向	機 能	端子 番号	信号名	方向	機 能	信号名	方向	機能
A ₀₁	GND			←			B ₀₁	GND			←		
A ₀₂	V ₁			←			B ₀₂	V ₁			←		
A ₀₃	V ₂			←			B ₀₃	V ₂			←		
A ₀₄	AB ₀₀₁	I/O	アドレス・バス	←			B ₀₄	DB ₀₀₁	I/O	データ・バス	←		
A ₀₅	AB ₀₁₁	I/O	アドレス・バス	←			B ₀₅	DB ₀₁₁	I/O	データ・バス	←		
A ₀₆	AB ₀₂₁	I/O	アドレス・バス	←			B ₀₆	DB ₀₂₁	I/O	データ・バス	←		
A ₀₇	AB ₀₃₁	I/O	アドレス・バス	←			B ₀₇	DB ₀₃₁	I/O	データ・バス	←		
A ₀₈	AB ₀₄₁	I/O	アドレス・バス	←			B ₀₈	DB ₀₄₁	I/O	データ・バス	←		
A ₀₉	AB ₀₅₁	I/O	アドレス・バス	←			B ₀₉	DB ₀₅₁	I/O	データ・バス	←		
A ₁₀	AB ₀₆₁	I/O	アドレス・バス	←			B ₁₀	DB ₀₆₁	I/O	データ・バス	←		
A ₁₁	GND			←			B ₁₁	GND			←		
A ₁₂	AB ₀₇₁	I/O	アドレス・バス	←			B ₁₂	DB ₀₇₁	I/O	データ・バス	←		
A ₁₃	AB ₀₈₁	I/O	アドレス・バス	←			B ₁₃	DB ₀₈₁	I/O	データ・バス	←		
A ₁₄	AB ₀₉₁	I/O	アドレス・バス	←			B ₁₄	DB ₀₉₁	I/O	データ・バス	←		
A ₁₅	AB ₁₀₁	I/O	アドレス・バス	←			B ₁₅	DB ₁₀₁	I/O	データ・バス	←		
A ₁₆	AB ₁₁₁	I/O	アドレス・バス	←			B ₁₆	DB ₁₁₁	I/O	データ・バス	←		
A ₁₇	AB ₁₂₁	I/O	アドレス・バス	←			B ₁₇	DB ₁₂₁	I/O	データ・バス	←		
A ₁₈	AB ₁₃₁	I/O	アドレス・バス	←			B ₁₈	DB ₁₃₁	I/O	データ・バス	←		
A ₁₉	AB ₁₄₁	I/O	アドレス・バス	←			B ₁₉	DB ₁₄₁	I/O	データ・バス	←		
A ₂₀	AB ₁₅₁	I/O	アドレス・バス	←			B ₂₀	DB ₁₅₁	I/O	データ・バス	←		
A ₂₁	GND			←			B ₂₁	GND			←		
A ₂₂	AB ₁₆₁	I/O	アドレス・バス	←			B ₂₂	+12 V			←		
A ₂₃	AB ₁₇₁	I/O	アドレス・バス	←			B ₂₃	+12 V			←		
A ₂₄	AB ₁₈₁	I/O	アドレス・バス	←			B ₂₄	IR ₃₁	I	INT ₀	←		
A ₂₅	AB ₁₉₁	I/O	アドレス・バス	←			B ₂₅	IR ₅₁	I	INT ₁	←		
A ₂₆	AB ₂₀₁	I/O	未使用	AB ₂₀₁	I/O	アドレスバス	B ₂₆	IR ₆₁	I	INT ₂	←		
A ₂₇	AB ₂₁₁	I/O	未使用	AB ₂₁₁	I/O	アドレスバス	B ₂₇	IR ₉₁	I	INT ₃	←		
A ₂₈	AB ₂₂₁	I/O	未使用	AB ₂₂₁	I/O	アドレスバス	B ₂₈	IR ₁₀₁ /IR ₁₁₁	I	INT ₄ /INT ₄₍₁₎	IR ₁₀₁	I	INT ₄₁
A ₂₉	AB ₂₃₁	I/O	未使用	AB ₂₃₁	I/O	アドレスバス	B ₂₉	IR ₁₂₁	I	INT ₅	←		
A ₃₀	INT ₀	O		←			B ₃₀	IR ₁₃₁	I	INT ₆	←		
A ₃₁	GND			←			B ₃₁	GND			←		
A ₃₂	IOCHK ₀	I	外部 NMI (2)	←			B ₃₂	-12 V			←		
A ₃₃	IOR ₀	I/O	コマンド	←			B ₃₃	-12 V			←		
A ₃₄	IOW ₀	I/O	コマンド	←			B ₃₄	RESET ₀	O	/RESET	←		
A ₃₅	MRC ₀	I/O	コマンド	←			B ₃₅	DACK ₀₀	O	HDC	←		
A ₃₆	MWC ₀	I/O	コマンド	←			B ₃₆	DACK ₃₀ / DACK ₂₀	O	AUX (1)	DACK ₃₀	O	
A ₃₇	S ₀₀ *	I/O	S ₀	INTA ₀	I/O	割り込み	B ₃₇	DRQ ₀₀	I	HDC	←		
A ₃₈	S ₁₀ *	I	S ₁	NOWAIT ₀	I		B ₃₈	DRQ ₃₀ /DRQ ₂₀	I	AUX (1)	DRQ ₃₀	I	
A ₃₉	S ₂₀ *	I	S ₂	SALE ₁	I/O	アドレスラッチ	B ₃₉	WORD ₀	I		←		
A ₄₀	LOCK *	I		MACS ₀	I		B ₄₀	CPKILL ₀ *	I		EXHRQ ₁₀	I	
A ₄₁	GND			←			B ₄₁	GND			←		
A ₄₂	CPUENB ₁₀	O		←			B ₄₂	RQGT ₀ *	I/O	バスの解放要求	EXHLA ₁₀	O	
A ₄₃	RFSH ₀	O	リフレッシュ信号	←			B ₄₃	DMATC ₀	O	END OF PROCESS	←		
A ₄₄	BHE ₀	I/O		←			B ₄₄	NMI ₀	O		←		
A ₄₅	IRDY ₁	I	レディ信号	←			B ₄₅	MWE ₀	I/O		←		
A ₄₆	SCLK ₁	O	システム・クロック	←			B ₄₆	HLDA ₀₀ *	O		EXHLA ₃₀	O	
A ₄₇	SI8CLK ₁	O	307.2 kHz	←			B ₄₇	HRQ ₀₀ *	I		EXHRQ ₀₀	I	
A ₄₈	POWER ₀	O	電源確定信号	←			B ₄₈	DMAHLD ₀ *	I		SUBSRQ ₁	O	
A ₄₉	+5 V			←			B ₄₉	+5 V			←		
A ₅₀	+5 V			←			B ₅₀	+5 V			←		

* 8086/V30 タイプ・バス

PC9801/E/F/M/U/VF/VM/UV/CV/UF/UR/PC98DO/DO*

PC9801VM21(スロット#1)/VX2(スロット#1) PC9801LV21(PC9801LV-8)

* 80286/80386/80486/Pentium タイプバス

PC9801VX21/UX/RA/RX/EX/ES/RS/T/DX/DS/DA/FA/FS/FX/US/BA/BX

PC9801VM21(スロット#2.3.4)/VX2(スロット#2.3.4) PC9801LS/LX(PC9801LV-8)

PC9821/Ap/As/Ae/Ce/Af PC98XA/XL/XL2/RL/GS

EPSON-PC DESKTOP EPSON-PC L-SLOT(A27-A29 は未接続, PC286L/LE/LF/LP は若干異なる)

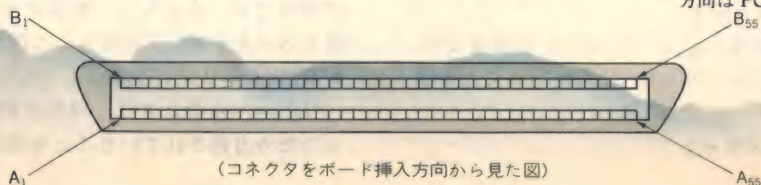
〈図 3-3 (b)-1〉 外部拡張コネクタ 110 ピン・タイプ(ノート・タイプ)

端子番号	信号名	方向	意 味	端子番号	信号名	方向	意 味
A ₁	+5 V	—	+5 V 電源	B ₁	+5 V	—	+5 V 電源
A ₂	+5 V	—	+5 V 電源	B ₂	SD ₁₅	I/O	データ・バス
A ₃	SD ₁₄	I/O	データ・バス	B ₃	SD ₁₃	I/O	データ・バス
A ₄	SA ₁₂	I/O	データ・バス	B ₄	SD ₁₁	I/O	データ・バス
A ₅	SA ₁₀	I/O	データ・バス	B ₅	SD ₉	I/O	データ・バス
A ₆	SA ₈	I/O	データ・バス	B ₆	SD ₇	I/O	データ・バス
A ₇	SA ₆	I/O	データ・バス	B ₇	SD ₅	I/O	データ・バス
A ₈	SA ₄	I/O	データ・バス	B ₈	SD ₃	I/O	データ・バス
A ₉	GND	—	グラウンド	B ₉	MFM	O	MFM 信号
A ₁₀	SD ₂	I/O	データ・バス	B ₁₀	SD ₁	I/O	データ・バス
A ₁₁	SD ₀	I/O	データ・バス	B ₁₁	SBHE	—	バス・ハイ・イネーブル
A ₁₂	SA ₁₉	O	アドレス・バス	B ₁₂	SA ₁₈	O	アドレス・バス
A ₁₃	SA ₁₇	O	アドレス・バス	B ₁₃	SA ₁₆	O	アドレス・バス
A ₁₄	SA ₁₅	O	アドレス・バス	B ₁₄	SA ₁₄	O	アドレス・バス
A ₁₅	SA ₁₃	O	アドレス・バス	B ₁₅	SA ₁₂	O	アドレス・バス
A ₁₆	SA ₁₁	O	アドレス・バス	B ₁₆	SA ₁₀	O	アドレス・バス
A ₁₇	GND	—	グラウンド	B ₁₇	SYNC	O	SYNC 信号
A ₁₈	SA ₉	O	アドレス・バス	B ₁₈	SA ₈	O	アドレス・バス
A ₁₉	SA ₇	O	アドレス・バス	B ₁₉	SA ₆	O	アドレス・バス
A ₂₀	SA ₅	O	アドレス・バス	B ₂₀	SA ₄	O	アドレス・バス
A ₂₁	SA ₃	O	アドレス・バス	B ₂₁	SA ₂	O	アドレス・バス
A ₂₂	SA ₁	O	アドレス・バス	B ₂₂	SA ₀	O	アドレス・バス
A ₂₃	GND	—	グラウンド	B ₂₃	GND	—	グラウンド
A ₂₄	SMRD	O	メモリ・リード・コマンド	B ₂₄	SMWR	O	メモリ・ライト・コマンド
A ₂₅	GND	—	グラウンド	B ₂₅	GND	—	グラウンド
A ₂₆	SIOR	O	I/O リード・コマンド	B ₂₆	SIOW	O	I/O ライト・コマンド
A ₂₇	NC	—	未接続	B ₂₇	IOCHK ₀	I	外部 NMI 要求信号
A ₂₈	INT ₅	I	INT ₅ (拡張用)	B ₂₈	INT ₃	I	INT ₃ (HDD)
A ₂₉	INT ₂	—	INT ₂ (未接続)	B ₂₉	DACK ₀₀	O	DMA アクノリッジ・チャンネル 0
A ₃₀	NC	—	未接続	B ₃₀	NC	—	未接続
A ₃₁	DACK ₃₀	O	DMA アクノリッジ・チャンネル 3	B ₃₁	DRQ ₀₀	I	DMA リクエスト・チャンネル 0
A ₃₂	INT ₄	—	INT ₄ (未接続)	B ₃₂	NC	—	未接続
A ₃₃	DRQ ₃₀	I	DMA リクエスト・チャンネル 3	B ₃₃	INTA ₀	O	割り込みアクノリッジ信号
A ₃₄	WGATE	O	ライト・ゲート信号	B ₃₄	NC	—	未接続
A ₃₅	NC	—	未接続	B ₃₅	NC	—	未接続
A ₃₆	SALE	O	アドレス・ラッチ信号	B ₃₆	SA ₂₀	O	アドレス・バス
A ₃₇	SA ₂₂	O	アドレス・バス	B ₃₇	REST ₀	O	システム・リセット
A ₃₈	SCLK	O	システム・クロック	B ₃₈	S18CLK	O	約 307.2 kHz
A ₃₉	GND	—	グラウンド	B ₃₉	GND	—	グラウンド
A ₄₀	IORDY	I	レディ信号	B ₄₀	INT ₀	I	INT ₀ (拡張用)
A ₄₁	NC	—	未接続	B ₄₁	INT ₁	I	INT ₁ (拡張用)
A ₄₂	CPUE ₀	O	CPU 動作中	B ₄₂	WORD ₀	I	DMA ワード転送要求信号
A ₄₃	POWER ₀	O	電源確定信号	B ₄₃	RFSH ₀	O	リフレッシュ信号
A ₄₄	INT ₆	I	INT ₆	B ₄₄	DMATC ₀	O	DMA End of Process
A ₄₅	SA ₂₁	I	アドレス・バス	B ₄₅	HID /AG	O	ヘッドロード信号/アナログ緑色ビデオ信号
A ₄₆	NC	—	未接続	B ₄₆	NMI ₀	O	NMI 出力信号
A ₄₇	SA ₂₃	O	アドレス・バス	B ₄₇	DIR /AR	O	ディレクション信号/アナログ青色ビデオ信号
A ₄₈	STEP /VSYNC	O	ステップ信号/垂直同期信号	B ₄₈	WINDOW	I	ウィンドウ信号
A ₄₉	SSEL /HSYNC	O	サイド・セレクト/水平同期信号	B ₄₉	WDATA /AB	O	ライト・データ信号/アナログ赤色ビデオ信号
A ₅₀	DS ₂	—	ドライブ・セレクト 2	B ₅₀	GND	—	グラウンド
A ₅₁	NC	—	未接続	B ₅₁	NC	—	未接続
A ₅₂	RDATA	I	リード・データ	B ₅₂	READY	I	ドライブ・レディ
A ₅₃	RGBSEL	I	FDD/CRT 選択	B ₅₃	DS ₃	O	ドライブ・セレクト 3
A ₅₄	TRK ₀	I	トラック 0	B ₅₄	INDEX	I	インデックス信号
A ₅₅	WPRT	I	ライト・プロテクト	B ₅₅	SMWE ₀	O	メモリ・ライト・イネーブル

方向は PC 本体を基準としたもの

〈図 3-3 (b)-2〉

110 ピン・コネクタ



(コネクタをボード挿入方向から見た図)

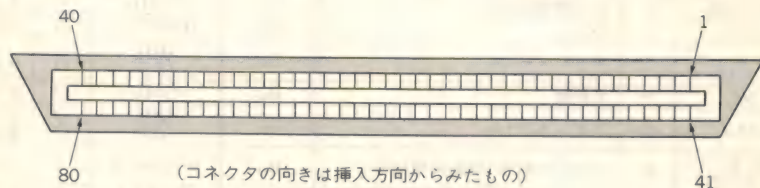
〈図 3-3 (b)-3〉 外部拡張コネクタ 80 ピン・タイプ (エプソンノート・タイプ)

端子番号	信号名	方向	意 味	端子番号	信号名	方向	意 味
1	SA ₁₂	I/O	アドレス・バス	41	SA ₁₉	I/O	アドレス・バス
2	SA ₁₁	I/O	アドレス・バス	42	GND	—	グラウンド
3	SA ₁₀	I/O	アドレス・バス	43	SA ₁₈	I/O	アドレス・バス
4	SA ₉	I/O	アドレス・バス	44	+5 V	—	+5 V 電源
5	SA ₈	I/O	アドレス・バス	45	SA ₁₇	I/O	アドレス・バス
6	SA ₇	I/O	アドレス・バス	46	INT ₀	I	INT ₀
7	SA ₆	I/O	アドレス・バス	47	SA ₁₆	I/O	アドレス・バス
8	SA ₅	I/O	アドレス・バス	48	INT ₁	I	INT ₁
9	SA ₄	I/O	アドレス・バス	49	SA ₁₅	I/O	アドレス・バス
10	SA ₃	I/O	アドレス・バス	50	GND	—	グラウンド
11	SA ₂	I/O	アドレス・バス	51	SA ₁₄	I/O	アドレス・バス
12	SA ₁	I/O	アドレス・バス	52	+5 V	—	+5 V 電源
13	SA ₀	I/O	アドレス・バス	53	SA ₁₃	I/O	アドレス・バス
14	SD ₁₅	I/O	データ・バス	54	INT ₂	I	INT ₂ *
15	SD ₁₄	I/O	データ・バス	55	SIOR	I/O	I/O リード・コマンド
16	SD ₁₃	I/O	データ・バス	56	INT ₃	I	INT ₃
17	SD ₁₂	I/O	データ・バス	57	SIOW	I/O	I/O ライト・コマンド
18	SD ₁₁	I/O	データ・バス	58	GND	—	グラウンド
19	SD ₁₀	I/O	データ・バス	59	SMRD	I/O	メモリ・リード・コマンド
20	SD ₉	I/O	データ・バス	60	+5 V	—	+5 V 電源
21	SD ₈	I/O	データ・バス	61	SMWR	I/O	メモリ・ライト・コマンド
22	SD ₇	I/O	データ・バス	62	DAK ₀₀	O	DMA アクノリッジ・チャンネル 0
23	SD ₆	I/O	データ・バス	63	CPUE ₀	O	CPU イネーブル
24	SD ₅	I/O	データ・バス	64	DRQ ₀₀	I	DMA リクエスト・チャンネル 0
25	SD ₄	I/O	データ・バス	65	IORDY	I	ウェイト要求信号
26	SD ₃	I/O	データ・バス	66	GND	—	グラウンド
27	SD ₂	I/O	データ・バス	67	SCLK	O	システム・クロック
28	SD ₁	I/O	データ・バス	68	+5 V	—	+5 V 電源
29	SD ₀	I/O	データ・バス	69	SBHE	I/O	バス・ハイ・イネーブル
30	DTCK	O	ドット・クロック (21.052 MHz)	70	DMATC ₀	O	DMA ターミナル・カウント
31	HSYNC	O	水平同期信号	71	SYNC	O	VSYNC と HSYNC の排他的論理和
32	VSYNC	O	垂直同期信号	72	(NC)	—	(未使用)
33	GRN ₁	O	表示データ緑 1	73	BLE ₁	O	表示データ青 1
34	RED ₁	O	表示データ赤 1	74	GND	—	グラウンド
35	GRN ₂	O	表示データ緑 2	75	BLE ₂	O	表示データ青 2
36	RED ₂	O	表示データ赤 2	76	+5 V	—	+5 V 電源
37	GRN ₃	O	表示データ緑 3	77	BLE ₃	O	表示データ青 3
38	RED ₃	O	表示データ赤 3	78	REST ₀	O	システム・リセット
39	GRN ₄	O	表示データ緑 4	79	BLE ₄	O	表示データ青 4
40	RED ₄	O	表示データ赤 4	80	GND	—	グラウンド

* PC-386NOTE W/WR は NC, 方向は PC 本体を基準としたもの

〈図 3-3 (b)-4〉

80ピン外部拡張コネクタ



I/O アクセス用のリード・ストロブ信号です。I/O リードするサイクルで“L”レベルになります。

▶ IOW₀: I/O ライト

I/O アクセス用のライト・ストロブ信号です。I/O ライトするサイクルで“L”レベルになります。

▶ MRC₀: メモリ・リード

メモリ・アクセス用のリード・ストロブ信号です。メモリ・リードするサイクルに“L”レベルになります。

▶ MWC₀: メモリ・ライト

メモリ・アクセス用のライト・ストロブ信号です。メモリ・ライトするサイクルで“L”レベルになります。

▶ MWE₀: メモリ・ライト・イネーブル

MWC₀よりも遅れたタイミングのライト・ストロブ信号です。主として、拡張メモリに対する DRAM 書き込みタイミング信号として使われます。

▶ RFSH₀: リフレッシュ

“L”レベルのときに、バスが DRAM のリフレッシュのため占有されていることを示します。このときは、

メモリに対して読み書きを行ってはいけません。

▶ **IR₃₁~IR₁₃₁**: 割り込み要求信号 (INT₀~INT₆)

外部から CPU に対してマスカブル割り込みをかけるときの信号です。8259 (割り込みコントローラ) に接続されていて処理されます。PC98 では、8259 がエッジ・トリガ・モードで使用されているために、割り込み要求信号が、“L” から “H” へ変化したときに割り込みがかかります。

IR₁₀₁, IR₁₁₁ はフロッピ・インターフェース用の割り込み要求信号で、同じ端子 (B₂₈) に割り当てられています。8086/V30 の機種ではスロット番号が一番大きいスロットに IR₁₁₁ が割り当てられ、その他のスロットには IR₁₀₁ が割り当てられています。80286 以降の機種 (PC98XA を除く) では、すべて IR₁₀₁ が割り当てられ、IR₁₁₁ は使用できません。

▶ **IOCHK₀**: NMI 要求信号

CPU に対してノン・マスカブル割り込み (NMI) をかけるための入力端子です。NMI はソフトウェアから割り込みを禁止できないハードウェア割り込みで、拡張メモリのパリティ・エラー検出に使用されます。

▶ **INT₀**: (マスカブル) インタラプト

IR₃₁~IR₁₃₁ までの割り込み要求信号に、割り込みコントローラ (8259) が応答したことを示します。

▶ **NMI₀**: (ノン・マスカブル) インタラプト

IOCHK₀ (NMI 要求信号) があったことを示します。

▶ **SCLK₁**: システム・クロック

拡張バスでの CPU のクロックです。8086 ではデュエティ比が 2:1 ですが、それ以外の CPU では 1:1 になっています。

CPU クロック 8/16 MHz=7.982 MHz

10/12/20 MHz=9.8304 MHz

5 MHz=4.9152 MHz

基本的には、上記のように CPU クロックで、システム・クロックが決まりますが、中には、CPU クロックとは無関係に決定されている機種もあります。

▶ **S18CLK₁**: (307.2 kHz)

307.2 kHz のクロック信号です。シリアル回線用のボーレート・クロックとして使用すると便利です。

▶ **POWER₀**: 電源確認信号

電源 ON/OFF 時に DC+5 V 電源電圧が、+4.75 V 以上になったときに有効になります。

▶ **RESET₀**: リセット信号

電源投入時に DC+5 V 電源電圧が、+4.75 V 以上になったときか、本体のリセット・スイッチが押されたときに、“L” レベルになります。

▶ **DRQ₀₀~DRQ₃₀**: DMA 要求信号

DMA によってデータ転送を行うときに、I/O デバイスが DMA コントローラに対して行う DMA 要求信号です。DMA コントローラは、この信号を見て、CPU にバス空け渡し (バス・ホールド) 信号を出します。

DRQ₂₀, DRQ₃₀ はフロッピ・インターフェース用の DMA 要求信号で、同じ端子 (B₃₈) に割り当てられています。前述の IR₁₀₁, IR₁₁₁ と同様に、バスの種類によって出ているスロット番号が違います。80286 以降の機種では DRQ₃₀ が割り当てられています。

▶ **DACK₀₀~DACK₃₀**: DMA アクノレッジ信号

DMA が出したバス・ホールド信号に CPU が許可を与えられたときに、DMA 要求をしたデバイスに、DMA が使用可能になったことを知らせる信号です。

DACK₂₀, DACK₃₀ はフロッピ・インターフェース用の DMA アクノレッジ信号で、同じ端子 (B₃₈) に割り当てられています。前述の IR₁₀₁, IR₁₁₁ と同様にバスの種類によって、出ているスロット番号が違います。80286 以降の機種では DACK₃₀ が割り当てられています。

▶ **WORD₀**: ワード/バイト

内部 DMA に接続する I/O デバイスがワード転送かバイト転送かを示す信号です。ワード転送のときに DACK 信号と同期させて、この信号を “L” にしなければなりません。ただし、ノーマル・モードでは未使用です。

▶ **DMTC₀**: DMA ターミナル・カウンタ

DMA 転送の転送時の終了バイトのときに “L” になります。

▶ **CPUENB₁₀**: CPU イネーブル信号

トランジスタ技術

SPECIAL No.35

特集 C言語による回路シミュレータの製作

Quick Cでのプログラミングとフィルタ回路の解析

MICRO-CAPIIIやPSPICEなどの回路シミュレータではどのようなアルゴリズムを使って電子回路を解析しているかを具体的に解説します。

好評発売中

B5判 160頁

定価1,600円 送料310円

CQ出版社 〒170 東京都豊島区巣鴨1-14-2 営業部 ☎03(5395)2141

〈図 3-4 (a)〉 バス・スロットのドライブ能力(8086/V30 CPU 機)

信号名	1 スロット当たりの 最大入力電流		外部ロジックがドライブ するときの最小出力電流(注)	
	I_{IL} (mA)	I_{IH} (μA)	I_{OL} (mA)	I_{OH} (mA)
AB ₁₀₁ ~AB ₁₉₁	-0.8	40	12	-1.2
BHE ₀				
DB ₀₀₁ ~DB ₁₅₁				
IOR ₀				
IOW ₀				
MRC ₀				
MWC ₀				
MWE ₀				
	不可			
RFSH ₀	全スロット で-0.8	全スロット で 40	12	-1.2
IR ₃₁ ~IR ₁₃₁	—	—	8	-0.4
IOCHK ₀	—	—		
INT ₀	-0.8	40	不可	
NMI ₀				
SCLK ₁				
S18CLK ₁				
POWER ₀				
RESET ₀				
DRQ ₀₀ , DRQ ₃₀	—	—	8	-0.4
DACK ₀₀ , DACK ₃₀	全スロット で-1.6*	全スロット で 80*	不可	
WORD ₀	—	—	不可	
DMAT ₀	全スロット で-1.6*	全スロット で 80*	8	-0.4
DMAHLD ₀	—	—		
HRQ ₀₀	—	—		
HRDA ₀₀	-0.8	40	不可	
CPUENB ₁₀				
IORDY ₀	—	—	8	-0.4
S ₀₀ , S ₁₀ , S ₂₀	全スロット で-0.4	全スロット で 20		
RQ/GT ₀	—	—		
LOCK ₀	全スロット で-0.4	全スロット で 20		
CPKILL ₀	—	—		

注：外部ロジックは IR₃₁~IR₁₃₁, DRQ₀₀, DRQ₃₀を除きトライ・ステート出力であること。トライ・ステート・ハイ・インピーダンス時のリーク電流は 20 μ A 以下とする

* PC9801 では、 I_{IL} =-0.8 mA, I_{IH} =40 μ A

CPU がバスを使用しているときに“L”になります。一般的には、アドレス・デコードのイネーブル端子に接続して、CPU アクセス時のみに動作するように使います。

► IORDY₁ : I/O レディ

CPU と内部 DMA に対するウェイト要求信号です。CPU に対して I/O デバイスのスピードがついてこないときに、“L”レベルにすることで CPU にウェイトがかかり、I/O アクセス時間を引き延ばします。通常は、CPU の速度に応じたウェイトが自動的に挿入さ

〈図 3-4 (b)〉 バス・スロットのドライブ能力(286 以降の CPU 機)

信号名	1 スロット当たりの 最大入力電流		外部ロジックがドライブ するときの最小出力電流(註)			
	I_{IL} (mA)	I_{IH} (μ A)	I_{OL} (mA)	I_{OH} (mA)		
AB ₀₀₁ ~AB ₂₃₁	-0.8	40	12	-1.2		
BHE ₀						
DB ₀₀₁ ~DB ₁₅₁						
IOR ₀		50				
IOW ₀						
MRC ₀						
NWC ₀		40				
MWE ₀						
RFSH ₀	不可					
IR ₃₁ ~IR ₁₃₁	—	—	8	-0.4		
IOCHK ₀	—	—				
INT ₀	-0.8	40	不可			
NMI ₀			—	—		
SCLK ₁			不可			
S18CLK ₁						
POWER ₀						
RESET ₀						
DRQ ₀₀ , DRQ ₃₀	—	—	8	-0.4		
DACK ₀₀ , DACK ₃₀	-0.8	40	不可			
WORD ₀			8	-0.4		
DMATC ₀	-0.4	20	不可			
CPUENB ₁₀	-0.8	40	12	-1.2		
IORDY ₁	—	—	8	-0.4		
EXHRQ ₁₀ , EXHRQ ₂₀	—	—				
EXHLA ₁₀ , EXHLA ₂₀	-0.8	40			不可	
SBUSRQ ₁						
NOWAIT ₀	—	—	8	-0.4		
SALE ₁	-0.8	50	12	-1.2		
INTA ₀		40	8	-0.4		
MACS ₀	—	—				

注：外部ロジックは、IR₃₁, IOCHK₀, DRQ₀₀, DRQ₃₀, EXHRQ₁₀, EXHRQ₂₀, NOWAIT₀, MACS₀がオープン・コレクタで、他はトライ・ステート出力であること

れます。IORDY 信号の“L”レベルの信号幅は最大 7 μ s 以下にします。

► GND : グラウンド

► +5 V : +5 V 電源ライン

► +12 V : +12 V 電源ライン

► V₁, V₂ : オプション電源ライン

オプション用の電源ラインは、本体側からは電源は供給されていません。また、すべての拡張スロットに接続されているので、他の拡張基板が V₁, V₂を使用していると、電源ラインがぶつかってしまう可能性があります。

以下の信号は、8086/V30 用のバス専用の信号で、

〈図 3-5〉 1 スロット当たりの電源容量

● PC9801/E/F/M/U/VF/VM/UV

DC	変動率	1 スロット当たりの容量
+ 5 V	± 5 %以内	0.5 A
+12 V	±10 %以内	0.06 A
-12 V	±10 %以内	0.07 A

● 前記以外

DC	変動率	1 スロット当たりの容量
+ 5 V	± 5 %以内	0.8 A
+12 V	±10 %以内	0.06 A
-12 V	±10 %以内	0.07 A

80286 以降の CPU を使用したバスにはありません。また、PC9801LV21 に、PC9801LV-08(I/O 拡張ユニット)を使用したときは無効です。

▶ DMAHLD₀ : DMA ホールド

内部の DMA の動作をすべてイン・アクティブにし、内部 DMA が動作しないようにする信号です。外部 DMA 等を使用するときに使います。この信号は DRAM のリフレッシュを止めますので、長時間(140 クロック以上)アクティブにしてはいけません。

▶ HRQ₀₀ : ホールド・リクエスト信号

CPU にホールドを要求する信号です。CPU はホールド要求されるとウェイト状態になります。

▶ HLDA₀₀ : ホールド・アクノリッジ信号

CPU はホールド状態になったことを示す信号です。

▶ S₀₀, S₁₀, S₂₀ : CPU ステータス信号

CPU のマキシマム・モードにおける、S₀, S₁, S₂ のステータス信号がそのまま出力されています。

▶ RQGT₀ : リクエスト/グラウンド信号

CPU のマキシマム・モードにおける、RQ/GT 信号です。

▶ LOCK₀ : ロック信号

CPU のマキシマム・モードにおける、LOCK 信号です。

▶ CPKILL₀

内部 CPU をバスから切り離すための信号です。

● 80286 以降の機種における信号線

以下の信号は、80286 以降の CPU を使用したバス用の信号です。8086/V30 用バス専用の信号と入れ替わりで導入されました。

▶ EXHRQ₁₀, EXHRQ₂₀ : ホールド・リクエスト信号
外部 CPU/DMA からのバス要求信号です。

▶ EXHLA₁₀, EXHLA₂₀ : ホールド・アクノリッジ信号
外部 CPU/DMA へのアクノリッジです。

▶ SUBSRQ₁ : バス解放要求信号
内部 DMA、リフレッシュ制御回路から外部 CPU/DMA に対するバス解放要求信号です。

▶ NOWAIT₀ : ノーウェイト信号

メモリを 0 ウェイトで動かすときの要求信号です。

▶ SALE₁ : 上位アドレス・バス・ラッチ信号

アドレス・バス AB₁₇₁~AB₂₃₁ のラッチ要求信号です。

▶ INTA₀ : 外部 CPU データ要求信号

外部 CPU から、8259(割り込みコントローラ)に対するデータ要求信号です。

▶ MACS₀ : メモリ・ボード自己アクセス信号

オプションのメモリ・ボードが、自身に対してアクセスされていることを示すために出力する信号です。I/O 拡張ユニットで使用し、オープン・コレクタ出力とします。

この信号を定義していないメモリ・ボードの I/O 拡張ユニットでの動作は保証されません。

拡張スロットの電氣的仕様

各信号線のドライブ能力

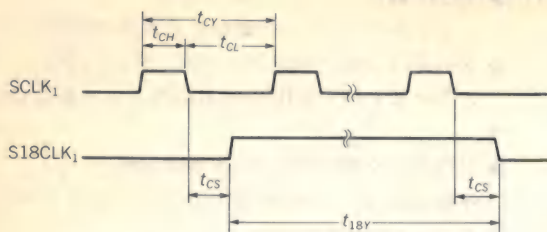
拡張スロットは、通常 2~4 個ありますが、例外的なものを除いて、バスはすべて並列に接続されています。拡張バスの入出力は、74F245(旧機種では 74LS245)等のバス・バッファ TTL でドライブされており、2~4 個の拡張基板を同時にドライブすることになります。

LS-TTL のドライブ能力(ファンアウト)は 20 個ほどですが、これは DC(直流)的な計算で、AC(交流)的にはもっと少なくなります。そのため、規定された拡張スロット一つ当たりの入力電流は決して多くありません。アドレス/データ・バスや、一般的なコントロール信号では、1 スロット当たりに許されている最大入力電流は -0.8 mA (I_{IL}) ですから、LS-TTL で換算すると 2 個までということになります。

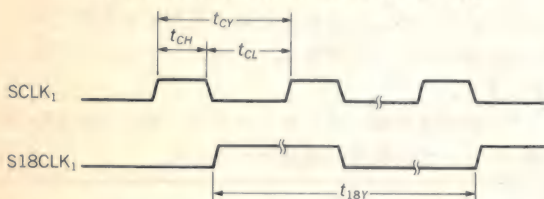
つまり、1 枚の拡張基板で、1 本のバス/制御線に二つの入力までつなぐことができます。また、TTL によっては、一つの入力で二つおんのファンインを持つものもありますので注意が必要です。

拡張基板の出力は、本体内部のバスをドライブするだけでなく、他の拡張基板の入力をもドライブすることになります。このため、拡張基板のドライブ電流は、最低 12 mA (I_{OL}) 必要と既定されています。これは、標準的な LS-TTL では満たすことができません。バッファ・タイプの LS-TTL (74LS245 等)を使用する必要があります。74HC245 等の CMOS バッファ ($I_{OL}=8$ mA) や、8255 等の LSI でも直接ドライブすることはできません。ただし、バスの制御信号には (I_{OL}) は 8 mA で良いものもあるので、これらは標準的な LS-TTL 等でもドライブ可能です。

〈図 3-6〉 8086 のシステム・クロック



〈図 3-7〉 V30 以降の CPU のシステム・クロック



記号	パラメータ	8 MHz モード (ns)		5 MHz モード (ns)	
t_{CY}	SCLK Cycle Time	125.20		203.45	
記号	パラメータ	min (ns)	max (ns)	min (ns)	max (ns)
t_{CH}	SCLK High Time	43	57	70	84
t_{CL}	SCLK Low Time	68	82	120	134
t_{18V}	S18CLK Cycle Time	307.200 (kHz)		307.200 (kHz)	
t_{CS}	S18CLK Delay Time	83	133	19	67

記号	486/Pentium*1		80286/386*2						70116			
			8/16 MHz		10/12/20 MHz		8MHz		10 MHz			
	min	max	min	max	min	max	min	max	min	max		
t_{CY}	101.73		125.20		101.73		125.20		101.73			
T_{CH}	45	56	58	68	45 (46)	56 (55)	46	69	38	56		
T_{CL}	45	56	58	68	45 (46)	56 (55)	56	79	46	64		
t_{18V}	307.20 (kHz)											

* 1 : PC9801FA は 80386 の 16 MHz と同じ値をとる

* 2 : PC9801US は 10/12/20 MHz の値をとる。()内は PC-98XL

80386 の 16 MHz 時は 8 MHz モードと同等のクロックが、80386 の 20 MHz 時は、10 MHz モードと同等のクロックが、80286 の 12 MHz 時は 10 MHz モードと同等のクロックが出力される

IOCHK₀や IORDY₁等の制御線は、バス上で並列に、ワイヤード OR されている可能性がありますので、これらの制御線ではオープン・コレクタの IC でドライブする必要があります。

バス・スロットのドライブ能力を図 3-4 に示します。

◆ 電源容量

拡張スロットには、電源として、+5 V、+12 V、-12 V の 3 種類の電源が供給されています。拡張基板 1 枚あたりに許されている電源容量は図 3-5 のようになります。比較的電流をたくさん使用する基板等には辛い規格のようで、基板外に電源を用意して別に供給する拡張基板も多いようです。

◆ 信号線タイミング

▶ システム・クロック (SCLK₁)

システム・クロックの周波数は、機種や CPU のクロック切り替えスイッチの位置によっても変わります。また、CPU が 8086 とそれ以外の CPU とでは、クロックのデューティ比が違います。

8086 のシステム・クロックを図 3-6 に、V30 以降の CPU のシステム・クロックを図 3-7 に示します。

8086/V30 用バスと 80286 以降の CPU 用バスでは、

タイミングが変更されているために、8086/V30 専用に設計されたメモリ・ボードは動かない可能性があります。両者の違いは以下のとおりになっています。

① アドレス・バスの上位 7 ビットは本体側でラッチされていないため、メモリ側でアドレス・デコードした後、SALE₁信号によりラッチする。また、自分のアドレスであった場合は、MACS₀信号を駆動する (MACS₀は拡張ユニットで使用)。

② メモリ・アクセスを 0 ウェイト (バス・サイクル 2 クロック) で動作させる場合は、NOWAIT₀信号を駆動すること。

③ DRAM に対するアクセスを行う場合は、ディレイド・ライト・サイクル・モードを使用すること。

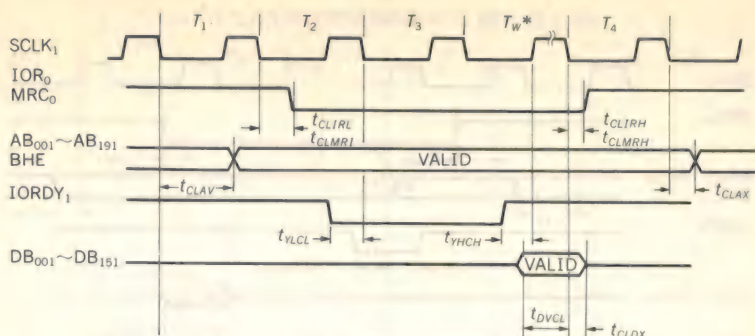
最近では、拡張メモリ・ボードの製作は、市販品よりコストもかかり自作する意味が薄くなってきました。

以下の 80286 以降の CPU でのアクセス・サイクルの解説は、拡張領域でのメモリ (主に ROM) アクセス・サイクルに限定して書いてあります。この領域でのアクセスは、自動的にウェイトが挿入されます。

▶ メモリ・リード・サイクル

メモリ・リード・サイクルは、8086/V30 では 4 ステート、80286 以降の CPU では 2 ステートと異なっています。しかし、メモリ・リードでメモリに求められるアクセス速度は、MRC₀がアクティブになってから、

〈図 3-8〉
8086 CPU I/O
MEMORY READ
タイミング



*I/O リード 5MHz-1wait 8MHz-2wait
メモリ・リード 5MHz-No wait 8MHz-1wait

CPU		8086-5 M	
記 号	パラメータ	min (ns)	max (ns)
t_{CLRL}	Command Active Delay	0	35
t_{CLRH}	Command Inactive Delay	0	35
t_{CLAV}	ADDRESS Valid Delay		128
t_{CLAX}	ADDRESS Hold Time	10	
t_{YCL}	IORDY Inactive Setup	87	
t_{YCH}	IORDY Active Setup	102	
t_{DVCL}	Read DATA Setup Time	54	
t_{CLDX}	Read DATA Hold Time	10	

CPU		8086			
記 号	パラメータ	8 MHz モード		5 MHz モード	
		min (ns)	max (ns)	min (ns)	max (ns)
t_{CLMRL}	MRD Active Delay	0	35	0	35
t_{CLMRH}	MRD Inactive Delay	0	35	0	35
t_{CLAV}	ADDRESS Valid Delay		78		128
t_{CLAX}	ADDRESS Hold Time	0		0	
t_{DVCL}	Read DATA Setup Time	40		54	
t_{CLDX}	Read DATA Hold Time	10		10	
t_{MLRH}	MRD Pulse Width	376 (3T)		407 (2T)	
t_{IRLH}	IOR Pulse Width	501 (4T)		610 (3T)	
t_{CLRL}	IOR Active Delay	80	138	14	74
t_{CLRH}	IOR Inactive Delay	14	74	14	74
t_{YCL}	IORDY Inactive Setup	159		237	
t_{YCH}	IORDY Active Setup	116		168	

読み出し CPU がデータ読み取りを行うまでのタイミングは基本的には似ています。

メモリに求められる速度は、8086/V30 の場合、

$$T_{acc} = (2+n) \times T_{cy} - T_{CLMRL} - T_{DVCL}$$

$$(n: \text{CLOCK, } 5 \text{ MHz}=0, 8 \text{ MHz}=1, 10 \text{ MHz}=1)$$

で算出でき、80286 以降の CPU では、 t_{mRDS} で規定されます。

▶ I/O リード・サイクル

I/O リード・サイクルは、8086 と V30 で、 IOR_0 のアクティブになるタイミングが違います。もちろん 80286 以降の CPU も、メモリ・リード・サイクル同様にステート数が違います。

I/O デバイスに求められる速度は、8086 の場合はメモリ・リード・サイクルと同等な算出方法で、

$$T_{acc} = (2+n) \times T_{cy} - T_{CLIRL} - T_{DVCL}$$

$$(n: \text{CLOCK, } 5 \text{ MHz}=1, 8 \text{ MHz}=2)$$

で算出でき、V30 の場合は、

$$T_{acc} = (1+n) \times T_{cy} - T_{CLIRL} - T_{DVCL}$$

$$(n: \text{CLOCK, } 8 \text{ MHz}=2, 10 \text{ MHz}=3)$$

となります。80286 以降の CPU では、 T_{IORDS} で規定さ

れます。

▶ メモリ・ライト・サイクル

メモリ・ライト・サイクルは、メモリ・リード・サイクルと同様なタイミングです。データ・バスの内容が確定したことを知るためには、 MWE_0 を使用します。

メモリに求められる速度は、8086/V30 の場合、

$$T_{acc} = (1+n) \times T_{cy} - T_{CLMWL} + T_{CLWH}$$

$$(n: \text{CLOCK, } 5 \text{ MHz}=0, 8 \text{ MHz}=1, 10 \text{ MHz}=1)$$

で算出できます。

▶ I/O・ライト・サイクル

I/O ライト・サイクルは、メモリ・ライト・サイクルにくらべて IOW_0 がアクティブになるタイミングが遅れます。V30 の I/O リード・サイクルに似ています。

I/O デバイスに求められる速度は、8086/V30 の場合、

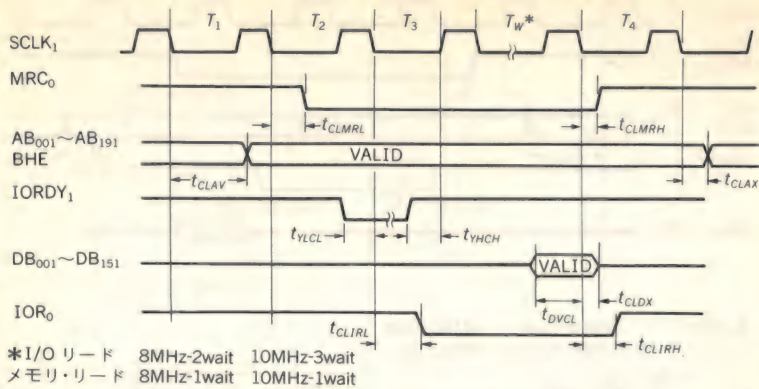
$$T_{acc} = (1+n) \times T_{cy} - T_{CLIL} + T_{CLIH}$$

$$(n: \text{CLOCK, } 5 \text{ MHz}=1, 8 \text{ MHz}=2, 10 \text{ MHz}=3)$$

で算出できます。

図 3-8 に 8086、図 3-9 に V30 の CPU IO/MEM

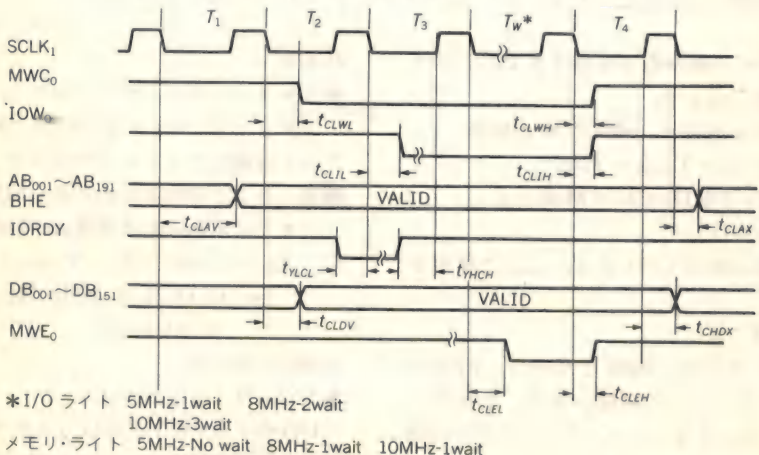
〈図 3-9〉 V30 CPU IO/MEMORY READ タイミング



CPU		70116 (V30)			
記号	パラメータ	8 MHz モード		10 MHz モード	
		min (ns)	max (ns)	min (ns)	max (ns)
t_{CLMRL}	MRD Active Delay	-45	38	-45	38
t_{CLMRH}	MRD Inactive Delay	0	35	0	35
t_{CLAV}	ADDRESS Valid Delay		78		68
t_{CLAX}	ADDRESS Hold Time	0		0	
t_{DVCL}	Read DATA Setup Time	38		28	
t_{CLDX}	Read DATA Hold Time	10		10	
$t_{MRLH}^{(*)}$	MRD Pulse Width	376 (3T)		305 (3T) (***)	
$t_{IRLH}^{(*)}$	IOR Pulse Width	361 (4T)		392 (5T)	
t_{CLIRL}	IOR Active Delay	-47	50	-47	50
t_{CLIRH}	IOR Inactive Delay	0	35	0	35
$t_{YLCL}^{(**)}$	IORDY Inactive Setup				
t_{YHCH}	IORDY Active Setup	60		43	

* : CPU 外部から Wait をかけないとき
** : IORDY はバスに対して非同期でよい。本規格値内でクロックに対して変化させれば、次の Cycle の動作が保証される
*** : オプション ROM のアドレス空間では 407 (4T) となる

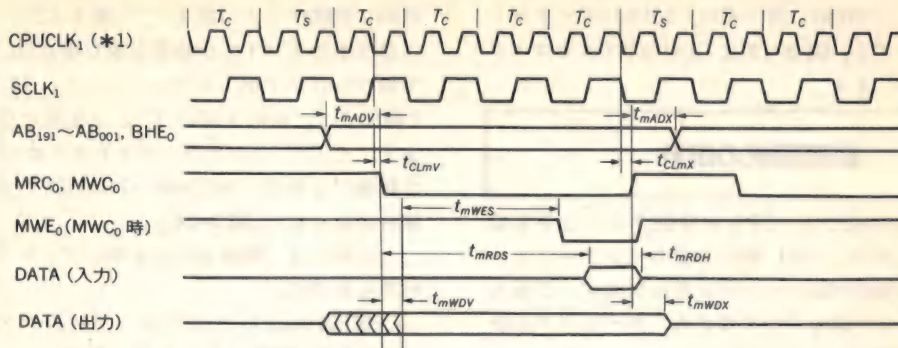
〈図 3-10〉 8086/V30 CPU IO/MEMORY WRITE タイミング



CPU		8086-5 MHz		8086		70116 (V30)	
CLOCK		5 MHz モード		8 MHz モード		5 MHz モード	
記号	パラメータ	min (ns) max (ns)		min (ns) max (ns)		min (ns) max (ns)	
		min (ns)	max (ns)	min (ns)	max (ns)	min (ns)	max (ns)
t_{CLWL}	MWC Active Delay	0	35	0	35	0	35
t_{CLWH}	MWC Inactive Delay	0	35	0	35	0	35
t_{CLIL}	IOW Active Delay	-70	75	43	74	-70	74
t_{CLIH}	IOW Inactive Delay	15	75	14	74	14	100
t_{CLDV}	Write Data Valid Delay		122		72		122
t_{CHDX}	Write Data Hold Time	10		10		10	
t_{CLEL}	MWE Active Delay	114	211	11	37	11	37
t_{CLEH}	MWE Inactive Delay	40	130	19	65	19	65

〈図 3-11〉 80286 以降 CPU MEMORY アクセス・タイミング

(0C000H~0DFFFFH, FC000H~FDFFFFH, 08000H~09FFFFH RAM KILL 時)



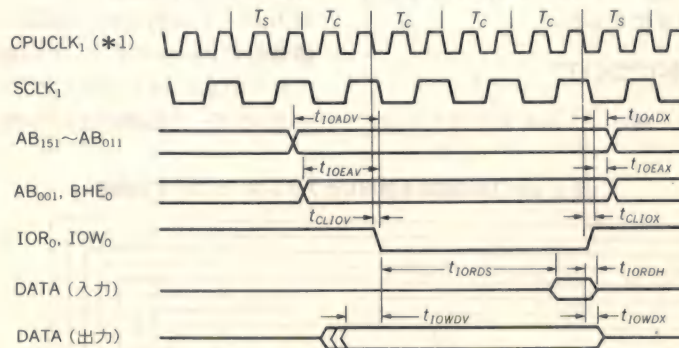
*1: CPUCLK₁ 信号は拡張バス上へは出力されない

記号	80286/386									
	8 MHz		10 MHz		12 MHz		16 MHz		20 MHz	
	min	max	min	max	min	max	min	max	min	max
t_{mADV}	50		50		40		50		50	
t_{mADX}	18		18		15		18		18	
t_{CLmV}	3	25	3	21			3	25	3	21
t_{CLmX}	3	25	3	20			3	25	3	20
t_{mWDS}	302		332		302		302		329	
t_{mRDS}		308		332		308		295		317
t_{mRDH}	5		5		5		5		5	
t_{mWDV}	-17		-55		-55		-17		-55	
t_{mWDX}	15		14		14		15		14	

注意: PC9801US は 20 MHz の値をとる

記 号	486/Pentium	
	min	max
t_{mADV}	50	
t_{mADX}	18	
t_{CLmV}	拡張バス上 I/O 拡張ユニット上	3 21 24 67
t_{CLmX}	拡張バス上 I/O 拡張ユニット上	3 20 12 45
t_{mWDS}	329	
t_{mRDS}		317
t_{mRDH}	5	
t_{mWDV}		-55
t_{mWDX}	14	1

〈図 3-12〉 80286 以降 CPU I/O アクセス・タイミング



*1: CPUCLK₁ 信号は拡張バス上へは出力されない

記号	80286/386									
	8 MHz		10 MHz		12 MHz		16 MHz		20 MHz	
	min	max	min	max	min	max	min	max	min	max
t_{IOADV}	115		139(150)		139		115		136	
t_{IOEAV}	111		139(150)		139		111		136	
t_{IOADX}	25		21(32)		21		25		21	
t_{IOEAX}	25		21(32)		21		25		21	
t_{CLIOV}	3	25	3	21			3	25	3	21
t_{CLIOX}	3	25	3	20			3	25	3	20
t_{IORDS}		270		254(312)		254		257		239
t_{IORDH}	5		5		5		5		5	
t_{IOWDV}	67		140(151)		130		67		140	
t_{IOWDX}	23		21(32)		21		23		21	

注意: PC9801US は 20 MHz の値をとる。()内は PC-98XL

記 号	486/Pentium	
	min	max
t_{IOADV}	136	
t_{IOEAV}	136	
t_{IOADX}	21	
t_{IOEAX}	21	
t_{CLIOV}	拡張バス上 I/O 拡張ユニット上	3 21 10 40
t_{CLIOX}	拡張バス上 I/O 拡張ユニット上	3 20 10 39
t_{IORDS}		239
t_{IORDH}	5	
t_{IOWDV}	140	
t_{IOWDX}	21	

ORY READ タイミングを、図 3-10 に 8086/V30 CPU IO/MEMORY WRITE タイミングを示し、また、図 3-11 に 80286 以降の CPU MEMORY アクセス・タイミングを、図 3-12 に CPU IO アクセス・タイミングを示します。

拡張基板の設計

拡張基板の作成では、CPU に接続することを前提に作られた LSI や、TTL 等の入出力インターフェースを使用する場合がほとんどだと思います。これらは、使用する信号線も少なくすみ、タイミングの計算なども比較的簡単にすみます。

特に 80 系の CPU への接続を前提に作られた LSI (μ PD8255 等)は、極めて簡単に接続できるようになっています。

■ I/O アドレスについて

ユーザに開放されている I/O ポート・アドレスは、 $\times nDOH \sim \times nEFH$ で、 n は 0~7 までです。

$n=8 \sim F$ までは NEC のリザーブになっていますから、今後、拡張基板を設計する場合は最低でも 12 ビット、できれば 16 ビット(フル)デコードする必要があると思われます。

他の拡張基板でも、この領域を使用している基板は数多くありますので、それらとかわち合わないためにも、I/O アドレスは固定せずに、ディップ・スイッチ等で変更可能な設計にするべきでしょう。

■ アドレス・デコードについて

アドレス・デコードを、何ビットぶんまで行うかは、

コストや部品点数に関係してきます。16 ビット・フル・デコードするなら、74LS688 という TTL を 2 個直列に接続するのが簡単です(図 3-13)。

基板上のデバイスが複数必要な場合は、74LS138 等で振り分けますが、アドレス・デコードに 74LS688 を 2 個使うと、合計 3 個の TTL が必要になってしまいます。そこでフル・デコードをあきらめ 12 ビット程度で我慢できれば、74LS688+74LS138 だけでも十分な場合があります(図 3-14)。

この例では、偶数番地に 8 個のアドレス・デコードが得られます。

CPUENB₁₀ という信号は、CPU がバスを使用しているときに“L”になります。アドレス・デコードの際に、いっしょにデコードさせておきます。

■ ワード・アクセスとバイト・アクセス

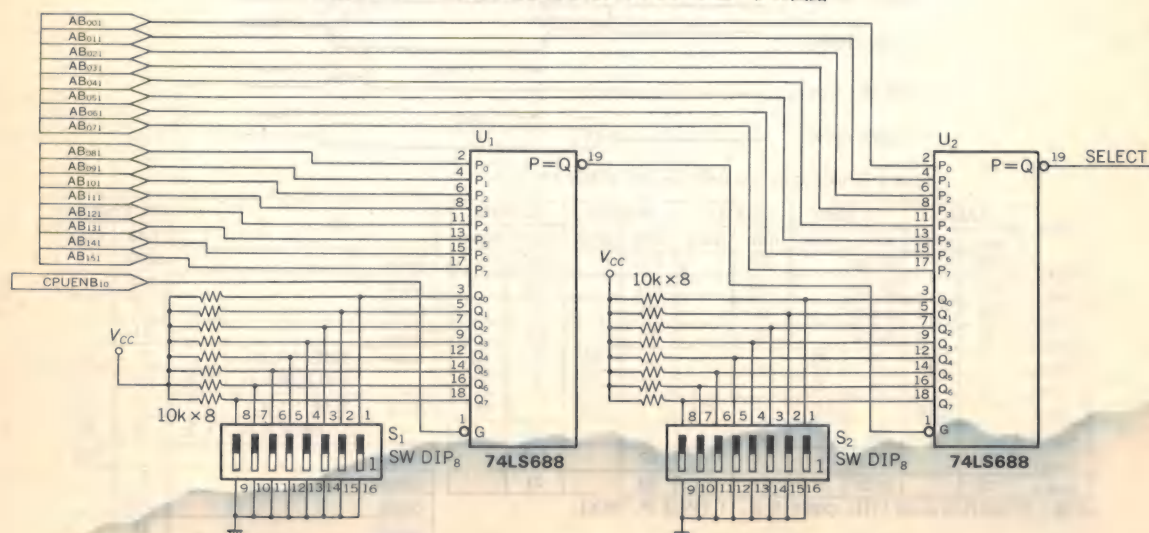
PC98 シリーズでも数多く使用されている 80 系周辺 LSI は、データ・バスの幅が 8 ビット(バイト)ですから、バイト・アクセスになります。8086 等 16 ビット CPU は、偶数アドレスが下位 8 ビット、奇数アドレスが上位 8 ビットになりますから、これら 8 ビット・バスの LSI は奇数か、偶数のどちらかのアドレスに割り当てなくてはなりません。

そのためアドレスのデコードは、偶数アドレスに配置する場合は AB₀₀₁ を使用し、奇数アドレスの場合は BHE₀ を使用します。

図 3-15 は、偶数アドレスに 8 ビット(1 バイト)の入力ポートの例です。SELECT には、図 3-14 の回路等のアドレス・デコード回路を接続します。

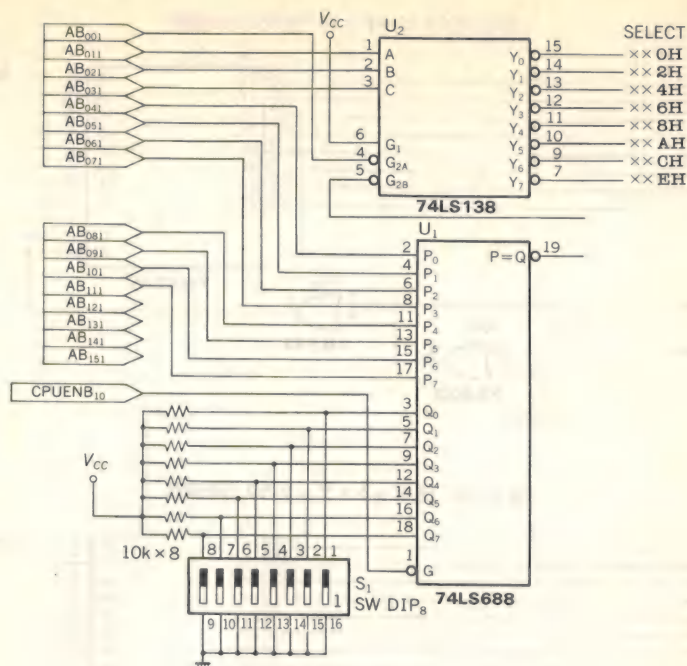
せっかくの 16 ビット CPU ですから、16 ビット(ワード)のデータ処理が可能な場合、16 ビットぶん一度

〈図 3-13〉 74LS688 を使用したアドレス・デコーダの回路



〈図 3-14〉

74LS688+74LS138
を使用したアドレス・
デコーダの回路



に入出力できると効率も上がって効果的です。この場合は、8ビットのポートが奇数、偶数の二つにあるという考え方で、アドレス・デコードをします。具体的には図3-16のようなデコード方法が良いでしょう。

このような方法ですと、偶数バイトだけや奇数バイトだけのアクセス時でも、またワード・アクセス時でも、ちゃんとアクセスできます。74LS244の代わりに74LS374等を使用すれば出力ポートにもできます。

■ バス・バッファについて

TTL等のICの出力の能力は有限です。拡張スロット等に使用されている74LS245等のTTLでは、出力が“L”(0V)のときに、24mAまでの電流に耐えられます。反対に、74LS00等の一般のTTLの入力を“L”にするためには、 -0.4mA 流す必要があります。つまり、74LS245の出力には、最大60個までの74LS00の入力をつなげることができます。これをファンアウト(出力)/ファンイン(入力)と呼びます。これはDC(直流)的な計算で、AC(交流)的にはもっと少なくなります。

PC98シリーズの拡張スロットにも、ファンアウト/ファンインが規定されています。アドレス/データ・バスや、一般的なコントロール信号では、許されている最大入力電流は、最大 -0.8mA ですから、74LS00等の標準的なTTLで換算すると2個までということになります。さらに、TTL(74LS266)によっては、一つの入力で二つぶんのファンインを持つものもありますので注意が必要です。

拡張基板のアドレス/データ・バスや、主要なコントロール信号の既定された出力電流は、最低12mA必要とされています。これは、74LS00等の標準的なLS-TTLの出力では満たすことができません。74LS245等のバッファ・タイプのLS-TTLを使用する必要があります。

図3-15のように、極めて簡単なTTL入出力インターフェースならば、バッファは必要ありませんが、入出力のポートが増えたり、複雑なロジックが必要になると、バスに2個以上のICがぶら下がることになり、別にバス・バッファが必要になってきます。また、LSI等のMOS系のデバイスの場合も、一般的には出力電流が多く取れないのでバス・バッファが必要になります。

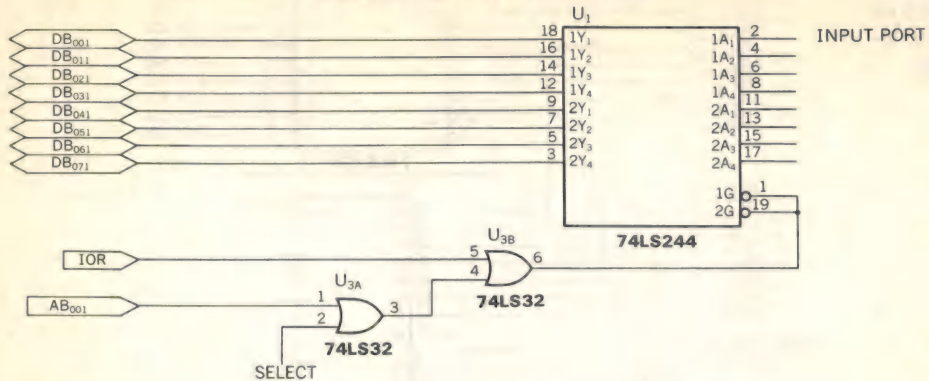
■ 入力ポートの設計

入力ポートのアクセス・タイミングについて考えてみましょう。

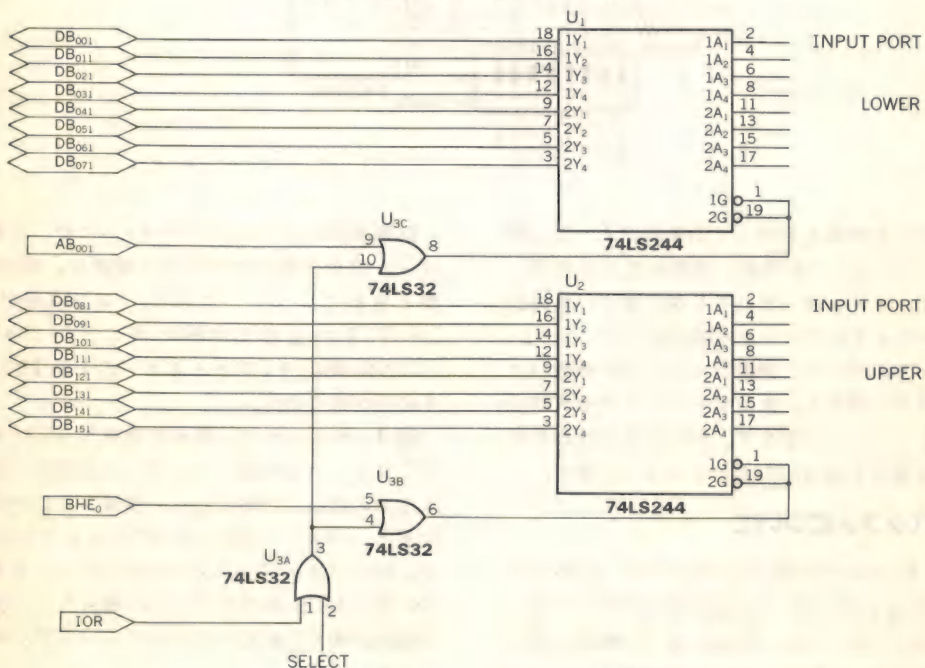
拡張基板のLSIやTTLからデータを読み込むときは、まずCPUが出力したI/Oアドレスが、自分のアドレスかどうか認識しなければなりません。そのときに使用するアドレス・デコーダは、アドレスが決定してから結果がわかるまで23ns(74LS688)の時間がかかります。アドレスをフル・デコードして、アドレス・デコーダが2段になっていれば、2倍の46nsかかる計算です。

次に、CPUの読み出し制御信号IOR₀が“L”になり、LSIやTTLが読み出しモードになります。実際

〈図 3-15〉 8 ビット・データの入力の回路



〈図 3-16〉 16 ビット・データの入力の回路



には、アドレスが確定してから 100 ns ほど経って IOR_0 が“L”になるので、先ほどのアドレス・デコーダの延長時間 (46 ns) は問題にはなりません。ただし、アドレス・デコードに複雑な回路を使い、100 ns を超えるようになると、タイミングを新たに考えなくてはなりませんので注意が必要です。

IOR_0 信号が“L”になると、アクセスされた LSI や TTL から CPU に向けてデータを出します。TTL 等では 30 ns (74LS541) 程度でデータが出てきますが、LSI では 100 ns 以上かかるのが普通です。その種類によってデータが出てくる時間が違うので、使用する LSI の仕様を調べておく必要があります。

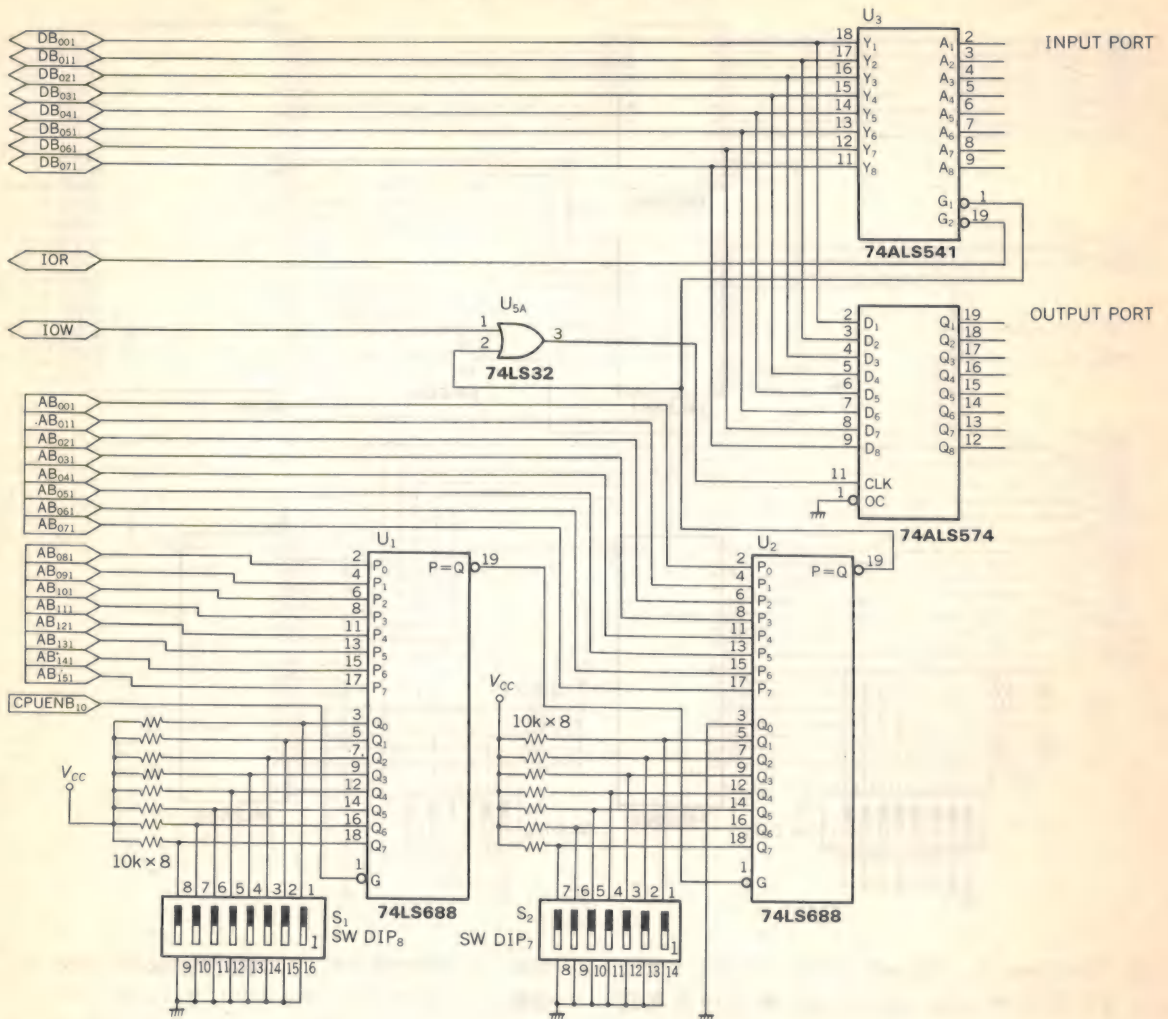
IOR_0 が“L”になってから、CPU が実際にデータ

を読み出すまでの時間は規定されています。最も速いタイミングを要求されるのは、CPU のクロックが 20 MHz のときで、239 ns となっています。前記のとおり TTL では 30 ns なので問題ありませんが、LSI では 239 ns より速くなくてはなりません。

さらに問題になるのは、バス・バッファを付けた場合です。一般的なバス・バッファ (74LS245) では、データがバッファを通り抜けるのに 12 ns ほどの延長時間があります。また、 IOR_0 信号にもバッファ (74LS244) が入っていると 18 ns ほど延長時間があるので、合わせて 30 ns ほど無駄に時間が浪費されてしまいます。

これから計算すると、LSI のアクセス・スピードは、

〈図 3-17〉 TTL を使用した入出力ポート



サンプル・プログラム・ディスク頒布のご案内

本誌に紹介されたサンプル・プログラムのソース・ファイルおよび実行用バイナリ・ファイルをディスク頒布します。

- 頒布価格 3,000 円 (送料, 税込み)
- 頒布メディア 3.5 インチまたは 5 インチ 2HD
(記入のない場合は 3.5 インチをお送りします)
- 申し込み期限 1994 年 10 月末日

● 申し込み方法

下記の申し込み用紙 (コピー可) に必要事項を記入のうえ, 代金を同封し現金書留でお申し込みください。

● 申し込み先

〒170 東京都豊島区巣鴨 1-14-2
CQ 出版(株)デザインウェーブ
ディスク・サービス TRSP45 係

● トランジスタ技術 SPECIAL No. 45 ソフトウェア申し込み用紙

送り先ご住所: 〒

お名前: _____

☎ () _____

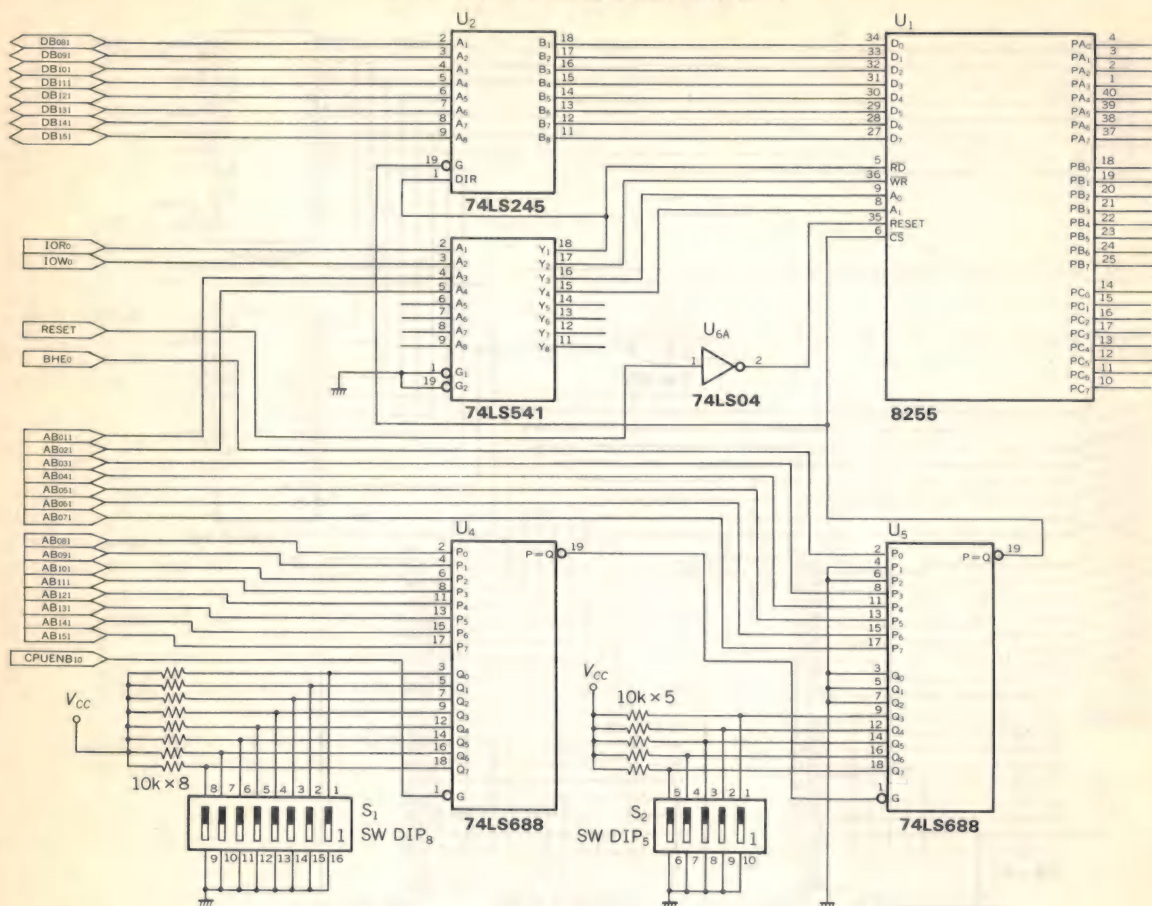
TRSP45

▶ 希望メディア (✓を入れてください)

☐ 5 インチ 2HD

☐ 3.5 インチ 2HD

〈図 3-18〉 μ PD71055 を使用した入出力ポート



239-30=209 ns で、209 ns 以下でなくてはなりません。よく使用される、8255 や 8251 等の 80 系周辺 LSI では、アクセス・スピードが 150~300 ns ほどなので、 μ PD8251AC-2 や、 μ PD71055 等の高速タイプ (160 ns) を使用しなくてはなりません。

◆ 出力ポートの設計

一般に、出力ポートのアクセス・タイミングは、入力ポートに比べて長めです。

出力の場合は、IOW₀信号が“L”になってから書き込みモードに入り、“H”になったときにデータを取り込みますので、IOW₀信号の長さが、アクセス・スピードになります。タイミング的には IOR₀より長くなります。

また、書き込まれるデータは、IOW₀が“L”になる前に確定していますから、多少のバス・バッファの延長には関係ありません。

◆ TTL の入出力ポートの設計

図 3-17 は、TTL だけで構成した 8 ビットの出入

力ポートの設計例です。この設計例では 74LS688 を 2 個使って、15 ビットをデコードしています。データ・バスが下位バイトに接続されているので、偶数アドレスを指定しないと動作しませんので、最下位ビットのアドレス・デコードは固定してあります。

◆ 80 系周辺 LSI の接続

図 3-18 は、 μ PD71055 (8255) を使用した入出力ポートの設計例です。この設計例では 74LS688 を 2 個使って、13 ビットをデコードしています。8255 が四つのアドレスを持つと、データ・バスが下位バイトに接続されているので、偶数アドレスを指定しないと動作しませんので、下位の 3 ビット分は固定されています。

また、8255 等の 80 系周辺 LSI では、RESET 入力 が正論理で、PC9801 シリーズの拡張スロットとは論理が反対になります。このために、74LS04 で反転する必要があります。

4

PC98シリーズのBIOS



PC9801 シリーズが持つ周辺ハードウェアの制御のほとんどは、BIOS を操作することで使用できます。この BIOS は、本体内蔵の ROM や拡張インターフェース基板上の ROM でサポートされます。BIOS のアクセスにはソフトウェア割り込み (INT 命令) が使用されていて、18H~1CH までが使用されています。

18H キーボード, CRT, グラフィック

19H RS-232-C

1AH プリンタ

1BH DISK

1CH カレンダー時計, タイマ

各 BIOS の呼び出しは、AH レジスタに機能番号を入れて、対応する INT 命令を実行することで、BIOS を振り分けています。このため、五つの割り込みだけでも多くの機能を処理できます。

また、BIOS へ受け渡すパラメータは、主にレジスタを使用しています。多量のパラメータを受け渡す場合には、メモリ上にパラメータを置き、そのメモリの

アドレスをレジスタへ入れて呼び出す場合もあります。また、BIOS 処理で起こったエラーは、キャリ・フラグ (CF) の状態や AH レジスタの内容で知ることができます。

キーボード BIOS

キーボードからの割り込みを処理し、内部のバッファにキー・データを格納するモジュールと、プログラムからのキーボード入力要求を受けて、内部のバッファからキー・データを返すモジュールからできています。

キー・コードは、キーボードから送られてくるデータで、それぞれのキーに割り当てられたデータです。キー・データはキー・コードを元に作られた ASCII (JIS) コードで、ユーザ・プログラムを扱うデータです。

割り込み番号	機能番号	入力パラメータ	戻り値	機能
INT	AH			
18H	OOH	なし	AH=キー・コード AL=キー・データ	「キー・データの読み出し」 キー・バッファ先頭に格納されているキー情報をキー・データ・コードに変換して読み出す。 キー・バッファが空であれば、なにか入力されるまで待つ。 読み終わったキー情報はキー・バッファから失われる。
	01H	なし	AH=キー・コード AL=キー・データ BH=AX に読み出したデータの状態 OOH=無効 01H=有効	「キー・バッファ状態のセンス」 戻り値の AH, AL は「キー・データの読み出し」と同様であるが、キー・バッファの状態は変わらない。キー・バッファが空であれば「無効」を返す。 「キー・バッファ状態のセンス」でキー・データが有効であることを確認したうえで、「キー・データの読み出し」を実行すれば、キー・バッファが空であっても、BIOS 内部でキー入力待ちすることはない。

INT	AH	入力パラメータ	戻り値	機 能
18H	02H	なし	AL=シフト・キーの状態	「シフト・キー状態のセンス」 現在押されているシフト・キーの状態を調べる。
	03H	なし	なし	「キーボード・インターフェースの初期化」 キーボード・インターフェースで使用するμPD8251の初期化と、BIOSで 使用しているメモリ・エリアの初期化を行う。
	04H	AL=キー・コード・グループ番号 (00H~0FH)	AL=キー・コード・グループ内の八つの キー状態	「キー入力状態のセンス」 キーボード上のキーを16個のグループ に分けて、そのグループのそれぞれのキ ーが押されているか、離されているかを 調べる。 (キー・コード・グループは図4-1を参照)

〈図4-1〉 キー・コード・グループ

キー・コード・グループ \ ビット	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇
0	ESC	1 ! ヌ	2 " フ	3 # アア	4 \$ ウウ	5 % エ	6 & オ	7 ' ヤ
1	8 (ユ	9) ヨ	0 ワ	= ホ	^ へ	¥	BS	TAB
2	Q タ	W テ	ヨ イ	R ス	T カ	Y ン	U ナ	I ニ
3	O ラ	P セ	@ っ	[{ り	⌋	A チ	S ト	D シ
4	F ハ	G キ	H ワ	J マ	K ノ	L リ	; + レ	: * ケ
5] } ム	Z ツ	X サ	C ソ	V ヒ	B コ	N ミ	M モ
6	く ね	く ル	／ ? メ	ー ロ	SPACE	XFER	ROLL UP	ROLL DOWN
7	INS	DEL	↑	←	→	↓	HOME CLR	HELP
8	—	/	7	8	9	*	4	5
9	6	+	1	2	3	=	0	,
A	・	NFER	vf・1	vf・2	vf・3	vf・4	vf・5	
B							HOME	
C	STOP	COPY	f・1	f・2	f・3	f・4	f・5	f・6
D	f・7	f・8	f・9	f・10				
E	SHIFT	CAPS	カナ	GRPH	CTRL			
F								

CRT BIOS

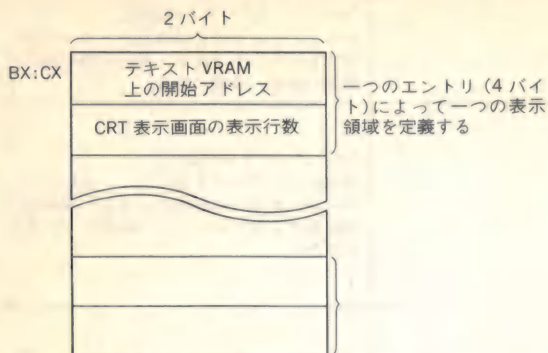
止や、設定・初期化等をできますが、文字出力・グラフィックの描画はできません。また、ブザーの制御もできます。

CRT 関連の制御を行います。画面表示等の開始・停

割り込み番号 INT	機能番号 AH	入力パラメータ	戻り値	機能																	
18H	0AH	AL=モード設定情報 <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td></td><td></td><td></td></tr></table> D ₀ = 画面当たりの行数 "0":25 行, "1":20 行 D ₁ = 行当たりの桁数 "0":80 字, "1":40 字 D ₂ =アトリビュート "0": パーチカル・ライン有効 "1": 簡易グラフ有効 D ₃ =KCG アクセス* "0": コード・アクセス有効 "1": ドット・アクセス有効 *: PC9801 では無効	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	0	0					なし	「CRT モードの設定」 テキスト表示用の GDC (μPD7220) 等の モード設定を行う。	
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀													
	0	0	0	0																	
	0BH	AL=なし	AL=モード設定状態 <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table> D ₀ = 画面当たりの行数 "1":20 行, "0":25 行 D ₁ = 行当たりの文字数 "1":40 字, "0":80 字 D ₂ =アトリビュート・タイプ "1": 簡易グラフ, "0": パーチカル・ライン D ₃ =KCG アクセス・タイプ* "1": ドット・アクセス, "0": コード・アクセス D ₇ =CRT の種類 "1": 専用高解像度ディスプレイ "0": 標準ディスプレイ *: PC9801 では無効	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀										「CRT モードのセンス」 「CRT モードの設定」で設定した情報 を読み出す。
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀													
	0CH	AL=なし	なし		「テキスト画面の表示開始」 テキスト画面表示用 GDC に表示開始要 求を行う。																
0DH	AL=なし	なし		「テキスト画面の表示停止」 テキスト画面表示用 GDC に表示停止要 求を行う。																	
0EH	DX=表示する領域の開始アドレス (下位 16 ビット指定する)	なし		「一つの表示領域の設定」 テキスト画面全体を一つの表示領域に設 定する。																	
0FH	BX=表示領域リストのセグメント・ アドレス CX=表示領域リストのオフセット・ アドレス DH=表示領域リストで最初に定義す るエントリの表示領域番号(1 ~3) 例えば、前回 3 分割しておき、 今回 2 番目の表示領域を変更す る場合、DH=1 とする DL=表示領域リストで定義するエン トリ個数(1~4) (図 4-2 参照)	なし		「複数の表示領域の設定」 テキスト画面を 4 まで分割し、それぞ れの表示領域に設定する。																	
10H	AL=カーソルを点滅にするか否かの 設定 OOH=点滅する OIH=点滅しない	なし		「カーソル・タイプの設定」 カーソルを点滅にするか否かを設定する。																	
	11H	なし	なし	「カーソルの表示開始」 カーソルを表示する。																	

INT	AH	入力パラメータ	戻り値	機能																										
18H	12H	なし	なし	「カーソルの表示停止」 カーソルを表示しない。																										
	13H	DX=カーソルを表示する位置の VRAM アドレス	なし	「カーソル位置の設定」 カーソルを表示する位置を設定する。																										
	14H	BX: CX=フォント・パターン・バッ ファの先頭アドレス ANK コードの場合 DL=展開するコード DH=CRT タイプ (OOH=標準 CRT 用 80H=専用高解像度 CRT 用) 漢字コードの場合 DX=展開するコード <div>下位アドレス 上位アドレス</div> <table><tr><td>制御域 2バイト</td><td>フォント・パターン・バッファ (8, 16, 32バイトのいずれか)</td></tr></table> <div>↑ BX: CX (フォント・パターン・ バッファの先頭アドレス)</div>	制御域 2バイト	フォント・パターン・バッファ (8, 16, 32バイトのいずれか)	なし	「フォント・パターンの読み出し(16 ド ット)」 設定されたフォントのパターンを、フォ ント・パターン・バッファへ読み出す。																								
	制御域 2バイト	フォント・パターン・バッファ (8, 16, 32バイトのいずれか)																												
	15H	なし	AH=ライト・ペンの状態 OOH=押されている O1H=押されていない DX=ライト・ペンが押された位置に 対応するテキスト VRAM のア ドレス	「ライト・ペン位置読み出し」 ライト・ペンが押されたかを通知する。 押された場合はその位置を知らせる。																										
	16H	DH=アトリビュートをクリアする文 字 DL=表示エリアをクリアする文字 <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>G</td><td>R</td><td>B</td><td>VL</td><td>UL</td><td>RV</td><td>BL</td><td>ST</td></tr></table> <div>D₀=シークレット D₁=プリンキング D₂=リバーズ D₃=アンダ・ライン D₄=バーチカル・ライン</div> <table><tr><td>カラー CRT</td><td>モノクロ CRT</td></tr><tr><td>D₅= 青</td><td>1 0 0</td></tr><tr><td>D₆= 赤</td><td>1 0 0 3bit による</td></tr><tr><td>D₇= 緑</td><td>1 1 0 濃淡表示</td></tr><tr><td></td><td>明 中 暗</td></tr></table>	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	G	R	B	VL	UL	RV	BL	ST	カラー CRT	モノクロ CRT	D ₅ = 青	1 0 0	D ₆ = 赤	1 0 0 3bit による	D ₇ = 緑	1 1 0 濃淡表示		明 中 暗	なし	「テキスト VRAM の初期化」 テキスト VRAM の全領域を指定された 文字でクリアする。
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																						
	G	R	B	VL	UL	RV	BL	ST																						
	カラー CRT	モノクロ CRT																												
	D ₅ = 青	1 0 0																												
D ₆ = 赤	1 0 0 3bit による																													
D ₇ = 緑	1 1 0 濃淡表示																													
	明 中 暗																													
17H	なし	なし	「ブザーの起呼」 ブザーを鳴らす。																											
18H	なし	なし	「ブザーの停止」 ブザーを停止。																											
19H	なし	なし	「ライト・ペン押下状態の初期化」 ライト・ペンが押された状態を検出する ための状態表示をクリアする。																											
1AH	BX: CX=フォント・パターン・バッ ファの先頭アドレス <div>下位アドレス(先頭) 上位アドレス</div> <table><tr><td>2バイト・ ワーク エリア</td><td>フォント・パターン 32バイト</td></tr></table> <div>↑(破壊される)</div> DX=登録コード	2バイト・ ワーク エリア	フォント・パターン 32バイト	なし	「ユーザ文字の定義(16 ドット)」 ユーザの作成した文字、記号のフォ ント・パターンを KCG RAM へ登録する。 登録された文字、記号は漢字と同様に、 テキスト VRAM に登録コードを書き込 むだけで表示される。																									
2バイト・ ワーク エリア	フォント・パターン 32バイト																													
1BH	AL=KCG アクセス・モードの設定 OOH=コード・アクセス O1H=ドット・アクセス	なし	「KCG アクセス・モードの設定」 KCG アクセス・モードを、コード・アク セスまたは、ドット・アクセスに設定す る。ドット・アクセスを選択すると、漢 字、ユーザ定義文字の表示ができなくな る。																											

〈図 4-2〉 表示領域リストの形式



テキスト VRAM 上の開始アドレス (2 バイト) :
GDC からみたアドレスを指定
(0000H ~ 1000H-1)
CRT 表示画面の表示行数 (2 バイト) : 20 行モードの場合 (1~20)
25 行モードの場合 (1~25)

グラフィック BIOS

GDC の操作だけで可能な、比較的簡単なグラフィックの制御が可能です。拡張グラフィック機能(4096 色中 16 色モード)等はサポートされていません。

グラフ BIOS 制御領域を図 4-3 に示します、

〈図 4-3〉 グラフ BIOS 制御領域

オフセット	ラベル	機 能	サイズ
0000	GBON_PTN	3 画面同時書き込み時の描画画面ナンバと描画オペレーション・モードの指定	RW ₁
0001	GBBCC	セットするボーダ・カラー・コード	RB ₁
0002	GBDOTU	単一画面の描画オペレーション・モードの指定	RB ₁
0003	GBDSP	描画開始方向	RB ₁
0004	GBCPC	パレット・レジスタにセットするカラー・コード	RB ₄
0008	GBSX1	描画開始アドレス X 座標	RW ₁
000A	GBSY1	描画開始アドレス Y 座標	RW ₁
000C	GBLNG1	書き込み長さ/第 1 描画方向ドット数	RW ₁
000E	GBWDPA	描画パターン・バッファの開始アドレス	RW ₁
0010	GBRBUF		RW ₃
0010	GBRBUF1	読み出しバッファ 1 の開始アドレス	
0012	GBRBUF2	読み出しバッファ 2 の開始アドレス	
0014	GBRBUF3	読み出しバッファ 3 の開始アドレス	
0016	GBSX2	描画終了アドレス X 座標	RW ₁
0018	GBSY2	描画終了アドレス Y 座標	RW ₁
001A	GBMDOT	マスキング・ドット数	RW ₁
001C	GBCIR	半径	RW ₁
001E	GBLNG2	第 2 描画方向ドット数	RW ₁
0020	GBLPTN	線種パターン	RW ₁
0020	GBMDOTI	基本パターン情報	RW ₈
0028	GBDTYP	描画タイプ	RW ₁
0029	GBFILL		RW ₁
002A-48	ワーク・エリア		

割り込み番号 INT	機能番号 AH	入力パラメータ	戻り値	機 能															
18H	40H	なし	なし	「グラフィック画面の表示開始」 グラフィック画面を表示する。															
	41H	なし	なし	「グラフィック画面の表示停止」 グラフィック画面を消す。															
	42H	CH=CRT ディスプレイのモード指定, VRAM 領域の指定(図 4-4 参照) <table border="1"><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table> <p>D₄=表示バンクの指定 "0": バンク 0 を使用, "1": バンク 1 を使用 D₅=CRT ディスプレイのモード指定 "0": カラー, "1": モノクロ D₆, D₇=VRAM 領域指定 "01":UPPER, "10":LOWER, "11":ALL</p> <p>注: PC9801/U2 では、表示バンクの指定は不可, "0" としておくこと</p>	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀					0	0	0	0	なし
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀												
				0	0	0	0												

INT	AH	入力パラメータ	戻り値	機能																	
18H	43H	DS: BX=UCW のアドレス UCW の GBCPC=パレット・レジスタにセットするカラー・コード(図 4-5 参照)	なし	「パレット・レジスタの設定」 パレット・レジスタにカラー・コードの設定を行う。モノクロ・モード時は表示画面の選択、合成になる。																	
	44H	DS: BX=UCW のアドレス UCW の GBBCC=セットするボーダ・カラー・コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>GBBCC</td><td>0</td><td>G</td><td>R</td><td>B</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	GBBCC	0	G	R	B	0	0	0	0		「ボーダ・カラーの設定」 標準ディスプレイを使用している場合はボーダ・カラーの設定ができる。専用高解像度ディスプレイでは、画面上下の色が出ない。
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀													
	GBBCC	0	G	R	B	0	0	0	0												
45H	CH=対象とする描画面面の指定 (図 4-6 参照) ES=描画パターン・バッファのセグメント・アドレス DS: BX=UCW のアドレス GBON __ PTN=3 画面同時書き込み時の描画面面ナンバと描画オペレーション・モードの指定 GBDOTU=単一画面の描画オペレーション・モードの指定 GBSX ₁ =描画開始アドレス X 座標 GBSY ₁ =描画開始アドレス Y 座標 GBLNG ₁ =書き込み長さ GBWDPA=描画パターン・バッファの開始アドレス	なし	「ドットの書き込み」 グラフィック画面にドットを描く、描画面面、描画バンクを指定できる。																		
46H	CH=対象とする描画面面の指定 ES=読み出しバッファ(1~3)のセグメント・アドレス DS: BX=UCW のアドレス 使用できる UCW GBSX ₁ =描画開始アドレス X 座標 GBSY ₁ =描画開始アドレス Y 座標 GBLNG ₁ =書き込み長さ GBRBUF ₁ =読み出しバッファ 1 の開始アドレス GBRBUF ₂ =読み出しバッファ 2 の開始アドレス GBRBUF ₃ =読み出しバッファ 3 の開始アドレス	なし <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table> D ₅ , D ₄ =描画面面ナンバ "00" P1 "01" P2 "10" P3 "11" P1/P2/P3 D ₆ =描画範囲 "0": LOWER/ALL "1": UPPER D ₇ =解像度モード "0": 標準解像度 "1": 専用高解像度	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀					0	0	0	0	「ドットの読み出し」 指定された描画面面から、読み出しバッファにドット単位の読み出しを行う。		
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀														
				0	0	0	0														
	47H	CH=対象とする描画面面の指定 (図 4-6 参照) DS: BX=UCW のアドレス GBON __ PTN=3 画面同時書き込み時の描画面面ナンバと描画オペレーション・モードの指定 GBDOTU=単一画面オペレーション・モードの指定 GBDSP=描画開始方向 GBSX ₁ =描画開始アドレス X 座標 GBSY ₁ =描画開始アドレス Y 座標 GBSX ₂ =描画終了アドレス X 座標 GBSY ₂ =描画終了アドレス Y 座標 GBLPTN=線種パターン GBDTYP=描画タイプ	なし	「直線、矩形の描画」 指定された描画面面に直線、矩形を書き込む。描画に使用される線種パターンも指定できる。																	

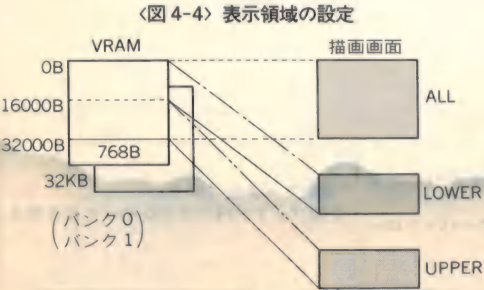
INT	AH	入力パラメータ	戻り値	機能
18H	48H	CH=対象とする描画面面の指定 (「直線、矩形の描画」と同じ) DS: BX=UCW のアドレス GBON __ PTN=3 画面同時書き込み 時の描画面面ナンバ と描画オペレーショ ン・モードの指定 GBDOTU=単一画面の描画オペレー ション・モードの指定 GBDSP=描画開始方向 GBSX ₁ =描画開始アドレス X 座標 GBSY ₁ =描画開始アドレス Y 座標 GBLNG=描画総ドット数 GBMDOT=マスキング・ドット数 GBCIR=半径 GBLPTN=線種パターン GBDTYP=描画タイプ	なし	「円弧の描画」 指定された描画面面に円弧を書き込む、 描画に使用される線種パターンも指定で きる。
	49H	CH=対象とする描画面面の指定 (「直線、矩形の描画」と同じ) DS: BX=UCW のアドレス GBON __ PTN=3 画面同時書き込み 時の描画面面ナンバ と描画オペレーショ ン・モードの指定 GBDOTU=単一画面の描画オペレー ション・モードの指定 GBDSP=描画開始方向 GBSX ₁ =描画開始アドレス X 座標 GBSY ₁ =描画開始アドレス Y 座標 GBLNG ₁ =第 1 描画方向ドット数 GBLNG ₂ =第 2 描画方向ドット数 GBMDOTI=基本パターン情報	なし	「グラフィック文字の描画」 指定された描画面面にグラフィック文字 を書き込む。
	4AH	CH=描画タイミング・モードの設定 06H=フラッシュ描画 16H=フラッシュレス描画	なし	「描画モードの設定」 描画面面に対する GDC からの書き込み タイミングの設定。 フラッシュ描画=表示期間と、 VRAM リフレッシュ期間を除いた時間 に、GDC が VRAM をアクセスできる。 フラッシュレス描画=表示期間でも GDC が VRAM をアクセスする。フラ ッシュ描画にくらべて約 5 倍の書き込み 速度が得られるが、画面にノイズが出る。

●使用上の注意

使用の前に、スタック・エリアをを 30 バイト確保し、SS、SP をセットする。

CPU のステータス・フラグのうち、IF(割り込み許可)=セット、TF(シングル・ステップ・モード)=クリアの状態にしておく。

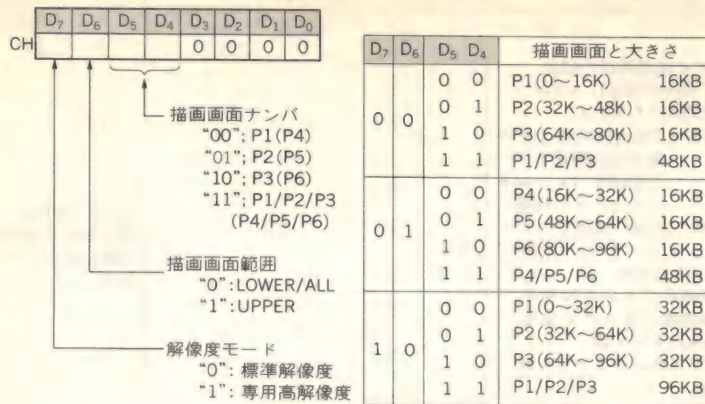
描画データ等の受け渡し、保存のために約 80 バイトの制御領域(UCW)を確保しなければならない。UCW はユーザの好きなアドレス(セグメント/オフセット)を指定できる。UCW の内容は以下図 4-3 のとおり。



〈図 4-5〉パレット・レジスタにセットするカラー・コード

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
#6	#7	#4	#5	#2	#3	#0	#1																								
0	G	R	B	0	G	R	B	0	G	R	B	0	G	R	B	0	G	R	B	0	G	R	B	0	G	R	B	0	G	R	B
1 バイト GBCPC								1 バイト								1 バイト								1 バイト							

〈図 4-6〉 ドットの描き込み



RS-232-C BIOS

本体に内蔵されている RS-232-C インターフェースのための BIOS で、拡張 RS-232-C インターフェースは拡張ボード上の ROM によってサポートされます。調歩同期式のための 75~9600bps までサポートされて

います。ブレイク信号の送受信はサポートされていません。

受信データは割り込み制御でバッファリングされますが、送信データはポーリングにより送出されます。受信は CTRL-S/CTRL-Q による X フロー制御が可能です。送信はいいいフロー制御は行われません (図 4-7, p.128)。

割り込み番号	機能番号	入力パラメータ	戻り値	機能
INT	AH			
19H	OOH	AL=トランスファ・レート (図 4-8 参照) BH=送信タイム・アウト時間 (01H~FFH・単位は 500ms) BL=受信タイム・アウト時間 (01H~FFH・単位は 500ms) CH= μ PD8251 のモード設定コマンド CL= μ PD8251 のコマンド指定 ES: DI=受信バッファのアドレス DX=受信バッファ・サイズ	AH=リターン・コード OOH=正常終了	「RS-232-C BIOS の初期化」 RS-232-C インターフェース、BIOS の初期化。
	01H	AL=トランスファ・レート (RS-232-C BIOS の初期化のトランスファ・レートを参照) BH=送信タイム・アウト時間 (01H~FFH・単位は 500ms) BL=受信タイム・アウト時間 (01H~FFH・単位は 500ms) CH= μ PD8251 のモード設定コマンド CL= μ PD8251 のコマンド指定 ES: DI=受信バッファのアドレス DX=受信バッファ・サイズ	AH=リターン・コード OOH=正常終了	「フロー制御を伴う初期化」 X パラメータでのフロー操作に対応した、RS-232-C インターフェース、BIOS の初期化。
	02H	なし	AH=リターン・コード OOH=正常終了 01H=RS-232-C の初期化が行われていない 02H=受信バッファがオーバ・フローした CX=受信データ長	「受信データ長の取得」 受信バッファ内の有効なデータ長を得る。

INT	AH	入力パラメータ	戻り値	機能																																
19H	03H	AL=送信データ	AH=リターン・コード OOH=正常終了 01H=RS-232-Cの初期化が行われていない 02H=受信バッファがオーバー・フローした 03H=送受信処理において、 μPD8251からの送受信可のステータスを引き取れなかった	「データの送信」 データを送信する。																																
	04H	なし	AH=リターン・コード OOH=正常終了 01H=RS-232-Cの初期化が行われていない 02H=受信バッファがオーバー・フローした 03H=送受信処理において、 μPD8251からの送受信可のステータスを引き取れなかった CH=受信データ CL=データ受信時のステータス	「データの受信」 受信され、バッファにためられたデータを読み出す。																																
	<div>ノーマル・モード</div> <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>DSR</td><td>BD</td><td>FE</td><td>OE</td><td>PE</td><td>TXE</td><td>CS</td><td>CD</td></tr></table> <div>μPD 8251 ステータス・バイト</div> <div>ハイレゾ・モード</div> <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>DSR</td><td>BD</td><td>FE</td><td>OE</td><td>PE</td><td>CI</td><td>CS</td><td>CD</td></tr></table> <div>μPD 8251 ステータス・バイト</div> <div>システム・ポート・ステータス</div> <div>モデム・ステータス</div>			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	DSR	BD	FE	OE	PE	TXE	CS	CD	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	DSR	BD	FE	OE	PE	CI	CS	CD	
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																												
DSR	BD	FE	OE	PE	TXE	CS	CD																													
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																													
DSR	BD	FE	OE	PE	CI	CS	CD																													
05H	CL=μPD8251へ出力するコマンド情報	AH=リターン・コード OOH=正常終了 01H=RS-232-Cの初期化が行われていない 02H=受信バッファがオーバー・フローした	「μPD8251へのコマンド出力」 μPD8251へ指示されたコマンドを出力する。																																	
06H	なし	AH=リターン・コード OOH=正常終了 01H=RS-232-Cの初期化が行われていない 02H=受信バッファがオーバー・フローした CH=μPD8251ステータス情報 CL=モデム・ステータス (図4-9参照)	「ステータスの取得」 μPD8251ステータスと、モデム・ステータスを得る																																	

〈図 4-8〉 転送レートの指定

トランスファ・レート (AL)	OOH	01H	02H	03H	04H	05H	06H	07H
伝送速度 (ボーレート, bps)	75	150	300	600	1200	2400	4800	9600

注：AL に 08H 以上の値を設定した場合は 1200bps とみなされる

〈図 4-9〉 ステータスの取得

ビット	略 称	1	0	ビット	略称	1	0
D ₇	DSR	Data Set Ready ON	Data Set Ready OFF	D ₇	\overline{CI}	着呼なし	着呼あり
D ₆	BD	ブレーク状態検出あり	ブレーク状態検出なし	D ₆	\overline{CS}	送信不可	送信可
D ₅	FE	フレーミング・エラー発生	フレーミング・エラーなし	D ₅	CD	受信キャリア検出なし	受信キャリア検出
D ₄	OE	オーバーラン・エラー発生	オーバーラン・エラーなし	D ₄	—	—	—
D ₃	PE	パリティ・エラー発生	パリティ・エラーなし	D ₃	—	—	—
D ₂	TXE	送信バッファ・エンプティ	送信バッファ・フル	D ₂	—	—	—
D ₁	RXRDY	受信レディ (受信キャラクタあり)	受信ビジー	D ₁	—	—	—
D ₀	TXRDY	送信レディ	送信ビジー	D ₀	—	—	—

CL=モデム・ステータス

注：PC9801 では、CI はサポートされていない

〈図 4-7〉 RS-232-C BIOS

		D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	
ES:DI→		INT	0	0	0	0	0	0	0	Interface Field:INT { “1”: データ受信の割り込みが発生した “0”: していない
+1		BFLG	0	0	0	0	0	0	0	BFLG: { “1”:BASIC で使用 “0”:BASIC で使用していない
バッファ・ コントロール・ ブロック	+2	INIT	BFULL	BOVF	XON	XOFF	0	0	0	(注) FLAG:RS-232-C BIOS 用内部フラグ・フィールド
	+3	/	IR	RTS	ER	SBRK	RXE	DTR	TXEN	
	+4									STIME : 送信時の TXRDY 待ち時間 (500ms 単位) セーブ領域
	+5									RTIME : 受信時の割り込み待ち時間 (500ms 単位) セーブ領域
	+6									XOFF:Buffer の格納可能データ長の 1/4 の値
	+7									
	+8									XON:Buffer の格納可能データ長の 3/4 の値
	+9									
	+0A									HEADP:Buffer の先頭アドレス
	+0B									
	+0C									TAILP:Buffer の最後尾 + 1 番地のアドレス
	+0D									
	+0E									CNT:Buffer 内の有効データ長(ワード単位)
	+0F									
+10									PUTP:Buffer の空エリアの先頭ポインタ	
+11										
+12									GETP:Buffer 内の有効データの先頭ポインタ	
+13										
受信バッファ 領域	+14	データ								
	+15	ステータス								
	+16	データ								
	+17	ステータス								
	+18									

*:BFLG :BASIC FLAG (COM1 用バッファには適用されない)
 FLAG : RS-232-C BIOS 用内部フラグ・フィールドの内容
 INIT : RS-232-C インタースェース (μPD8251) の初期化済み
 BFULL : 受信バッファが FULL
 BOVF : 受信バッファ・オーバーフローが発生
 XON : XON 処理 (CTRL-S 出力) を行う (3/4)
 XOFF : XOFF 処理 (CTRL-Q 出力) を行う (1/4)

プリンタ BIOS

ノーマル・モードのプリンタ・インターフェースは、

セントロニクス準拠で、フル・セントロニクス規格の簡易版になっています。印字データ出力は、I/O が終了するまで BIOS 内部でループします。

割り込み番号	機能番号	入力パラメータ	戻り値	機能
INT	AH			
1AH	10H	なし	AH=ステータス情報 OOH=プリンタ BUSY 01H=データ送信可	「プリンタ BIOS の初期化」 プリンタ・インターフェース、ステータス情報の初期化

INT	AH	入力パラメータ	戻り値	機 能
1AH	11H	AL=出力する1バイト・データ	AH=ステータス情報 OOH=プリンタ BUSY 01H=データ送信可 02H=タイム・アウト、データ未出力	「データの出力」 プリンタ・インターフェースが送信可能になるまで待つて、データを出力する。
	12H	なし	AH=ステータス情報 OOH=プリンタ BUSY 01H=データ送信可	「ステータスの取得」 現在のプリンタのステータス情報を取得する。
	30H	CX=データ長 ES: BX=データ・バッファのアドレス	AH=ステータス情報 OOH=プリンタ BUSY 02H=タイム・アウト、データ未出力	「データの出力(複数バイト)」 指定したバッファ上の、指定した長さのデータをプリンタに出力する。

DISK BIOS

DISK BIOS は、フロッピ・ディスクやハード・ディスクの区別なく INT 1BH で呼び出します。AH レジスタの機能コードで読み書き等の機能を指定し、AL レジスタにドライブ種別・ユニット番号を入れて、フロッピ・ディスク/ハード・ディスクの区別を行います。

DISK BIOS 機能一覧を図 4-10 に示します。

◆ 共通入力

DISK BIOS 呼び出しのときのパラメータは共通部分が多くなっていますので、まとめて解説します。

● AH=BIOS コマンド識別コード

コマンド(機能)は AH レジスタの下位ビット(0～3)で指定します。

上位ビット(4～7)はコマンド実行時の動作の指定をしますが、コマンドやデバイスにより受け付けられるパラメータが若干違います。

〈図 4-10〉 DISK BIOS の機能

AH レジスタ	機 能	1MBFD	640KBFD	1M, 640MB 両用 FD, RAM ドライブ	320KBFD	固定デ ィスク	SCSI
01H	ベリファイ	○	○	←	○	○	○
02H	診断のための読み出し	○	×	←	×	×	○
03H	初期化	○	○	←	○	○	○
04H	センス	○	○	○	○	○	○
05H	データの書き込み	○	○	←	○	○	○
06H	データの読み出し	○	○	←	○	○	○
07H	シリンダ 0 へのシーク	○	○	←	×	○	○
09H	テリィテッド・データの書き込み	○	×	←	×	×	○
0AH	ID の読み出し	○	○	←	×	×	○
0CH	テリィテッド・データの読み出し	○	×	←	×	×	○
0DH	トラックのフォーマット	○	○	←	○	○	○
0EH	動作モードの設定	×	×	○	○	×	×
0FH	リトラクト	×	×	×	×	○	○
10H	シーク	○	○	←	×	×	×
83H	モータ停止モードの設定	×	×	○	×	×	×
83H	初期化	×	×	○	×	×	×

注：←はアクセス・モードに応じて 1MBFD の機能や 640MBFD の機能が使えることを示す

〈図 4-11〉 BIOS コマンド識別コード (FDD)

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
MT	MF	r	SEEK	×	×	×	×

D₇(MT) : シングル・トラック "0"/ マルチトラック "1" の指定
(*同一シリンダの両面トラックのみ)

D₆(MF) : 単密度 (FM モード) "0"/ 倍密度 (MFM モード) "1" の読み出し指定

D₅(r) : 8 回リトライする "0"/ リトライなし "1" の指定

D₄(SEEK) : シーク動作を行う "1"/ 現在のトラック位置 "0" からの読み出し指定

コマンドに該当しないコードが指定された場合は正常終了 (CF="0") にする。

〈図 4-12〉 BIOS コマンド識別コード (HDD)

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
n	e	r	×	×	×	×	×

r=リトライ・フラグ (SASI のみ)

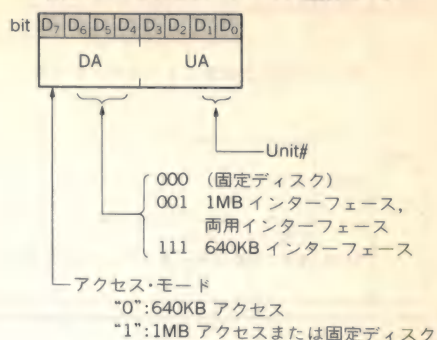
e=エラー通知ビット (SCSI のみ)

n=コマンドにより指定するものもある

〈図 4-13〉 アクセス・デバイスの種類の設定

デバイス	アクセス・デバイス	ユニット番号
0×H	SASI ドライブ(相対セクタ・アドレス)	0~6
1×H	1MB/640KB 両用ドライブでの 640KB モード	0~3
2×H	SCSI ドライブ(相対セクタ・アドレス)	0~6
5×H	320KB 専用ドライブ	0~3
7×H	640KB 専用ドライブ	0~3
8×H	SASI ドライブ(絶対セクタ・アドレス)	0~1
9×H	1MB(専用/両用を含む)ドライブ	0~3
A×H	SCSI ドライブ(絶対セクタ・アドレス)	0~6
F×H	640KB 専用ドライブ	0~3

〈図 4-14〉 アクセス・デバイス識別コード



▶ フロッピー・ディスク(1MB/640KB)

BIOS コマンド識別コード・FDD を図 4-11 に示しました。

▶ ハード・ディスク

BIOS コマンド識別コード・HDD を図 4-12 に示しました。

● AL=デバイス・タイプ識別コード

下位ビット(0~3)でユニット(ドライブ)番号を指定(UA)します。

上位ビット(4~7)でデバイスの種類指定(DA)します。

アクセス・デバイスの種類の指定を図 4-13 に、アクセス・デバイス識別コードを図 4-14 に示します。

● BX=データ長

フロッピー・ディスクの場合は、読み書きするデータの長さ(バイト)を指定します。

ハード・ディスクの場合は、256(512)バイト単位で指定します。

● CH=セクタ長

セクタ長は下記のように 0~3 の数字で表されます。ハード・ディスクや 320KB ドライブの場合は無効です。

CH	バイト数
0	128 バイト
1	256 バイト
2	512 バイト
3	1024 バイト

● CL(CX)=シリンダ番号

シリンダ番号で、デバイスによって下記のように指定範囲が違います。ハード・ディスクの場合は CX レジスタ(16 ビット)を使用します。

320KB 片面 0~34/両面 0~39

640KB 0~79

1MB 0~76

HDD 使用するドライブによって変わる

● DH=ヘッド番号

ヘッド番号を次のとおりに指定します。

FDD 0~1

HDD 使用するドライブによって変わる

● DL=セクタ番号

セクタ番号を指定します。

320KB 1~16

640KB 1~16

1MB 1~26

SASI 0~32(PC9801-27)

SCSI 使用するドライブによって変わる

● ES:BP=データ・バッファ領域先頭アドレス

読み書きするデータの先頭アドレスを指定します。

◆ 共通出力

▶ CF=終了条件

0=正常終了

1=異常終了

▶ AH=ステータス情報

◆ フロッピー・ディスク・ステータス情報一覧

ステータス情報一覧を図 4-15 に示します。

◆ ハード・ディスク・ステータス情報一覧

コマンド実行終了に得られるステータス情報は図 4-16 に示すような意味があります。

◆ 1MB/640KB 両用インターフェース

1MB/640KB 両用タイプのインターフェースは、1MB/640KB 自動切り替えインターフェース(1MB インターフェース・モード)と、640KB 専用インターフェース(640KB インターフェース・モード)の二つの動作モードがあります。

640KB インターフェース・モードは、640KB 専用インターフェースと同等の機能を持ち、DMA や割り込み等も同じになります。1MB インターフェース・モードでは、1MB 専用インターフェースの機能を含んだうえに、640KB のフロッピー・ディスクを扱う機能も持っています。

〈図 4-15〉 ステータス情報(FDD)

CF	AH		説 明		FDC STATUS との対応		詳 細
	16 進	ビット	略称	内 容	ST ₁₋₃	D _{0-D₅}	
0	00H	0000××××	NT	Normal end			DDAM(Deleted Data Address Mark)を検出した*1 媒体はセットされているが、ライトプロテクト状態 メモリ・アドレスがバンクにまたがるか、奇数番地から始まるように指定した 1回の動作の転送容量を越えてデータ長(DTL)を指定した デバイスから Fault 信号を受けとった*2,*3,*4 セクタ、メモリ間のデータ転送時に、一定時間内にデータ転送が終了できなかった*5 ユニットがノット・レディ状態 コマンド実行開始時、Write Protected 信号がオン
0	00H	0000××××	RY	Ready(センス)	ST ₃	D ₅	
0	10H	0001××××	CM	Control Mark (1MBFD)	ST ₂	D ₆	
0	10H	0001××××	WP	Write Protect (センス)	ST ₃	D ₆	
1	20H	0010××××	DB	DMA Boundary			
1	30H	0011××××	EN	ENd of cylinder	ST ₁	D ₇	
1	40H	0100××××	EC	Equipment Check	ST ₀	D ₇	
1	50H	0101××××	OR	OverRun	ST ₁	D ₄	
1	60H	0110××××	NR	Not Ready	ST ₀	D ₃	
1	70H	0111××××	NW	Not Writable	ST ₁	D ₁	
1	80H	1000××××	ER	ERror			ID 読み出し時に CRC エラーが発生した トラック内に、指定されたセクタが見つからなかった トラック内に、指定されたセクタが見つからず、かつ ID が 1 個もなかった、もしくは指定されたセクタの ID 検出後データ部のアドレス・マークを一定時間内(1ms/8MHz)に検出できなかった
1	90H	1001××××	TO	Time Out			
1	A0H	1010××××	DE	DataError(ID)	ST ₁	D ₅	
					ST ₂	D ₅	
1	BOH	1011××××	DD	DataError(Data)	ST ₁	D ₅	
					ST ₂	D ₅	
1	COH	1100××××	ND	No Data	ST ₁	D ₂	
1	DOH	1101××××	BC	Bad Sylinder	ST ₂	D ₁	
1	EOH	1110××××	MA	Missind Address mark (ID)	ST ₁	D ₀	
					ST ₂	D ₀	
1	FOH	1111××××	MD	Missind address mark (Data)	ST ₁	D ₀	
					ST ₂	D ₀	
0	01H	××××0001		両面媒体がセットされている	ST ₂ ST ₃	D ₀ D ₃	
1	08H	00001×××	CD	Corrected Data			
1	78H	00111×××	IA	Illegal disk Address			
1	88H	10001×××		Direct access an alternate track			
1	B8H	10111×××		Data Error			
1	C8H	11001×××		Seek error			
1	D8H	11011×××		代替トラックが読めない			

* 1 「データの読み出し AH=06H」ではそのセクタを読み出し後正常終了する。

* 2 「データの書き込み AH=05H」の場合は、データは書き込まれる。

* 3 「トラックのフォーマット AH=0DH」の場合は書き込み終了時にチェックされる。

* 4 「シリンダ 0 へのシーク AH=07H」の場合は一定時間内にトラック 0 を確認されなかった場合。

* 5 「データの書き込み AH=05H」の場合はそのセクタを書き込み後、OR となる。

ステータスのビット 7~5 は、CF=0 のとき "000"、CF=1 のとき "000" 以外となる。

ビット 3~0 は、センス・コマンドで装置種別が通知される (CF=0 のとき有効)。

ビット 0 { "0": 片面媒体がセットされている

"1": 両面媒体がセットされている (1MB/640KB 両用ドライブでは常に両面)

ビット 2 { "0": 40 シリンダ・モード、もしくは 1MFD アクセス・モード

"1": 80 シリンダ・モード

〈図 4-16〉 ステータス情報(HDD)

エラー・ステータス情報										リトライ処理
略称	内容	AH の内容								
NT	Normal end	0	0	0	0	×	×	×	×	
DB	DMA Boundary	0	0	1	0	×	×	×	×	コマンド使用上に誤りがある
EN	End of cylinder	0	0	1	1	×	×	×	×	
EC	Equipment Check	0	1	0	0	×	×	×	×	再試行する
OR	Over Run	0	1	0	1	×	×	×	×	
NR	Not Ready	0	1	1	0	×	×	×	×	
NW	Not Writable	0	1	1	1	×	×	×	×	
TO	Time Out	1	0	0	1	×	×	×	×	
DE	Data Error (ID)	1	0	1	0	×	×	×	×	Recalibrate → Seek
DD	Data Error (Data)	1	0	1	1	×	×	×	×	
ND	No Data	1	1	0	0	×	×	×	×	→リード/ライト系コマンド
MA	Missing Address mark (ID)	1	1	1	0	×	×	×	×	
MD	Missing Address mark (Data)	1	1	1	1	×	×	×	×	
IA	Illegal Disk Error	1	0	0	1	1	×	×	×	
	Defect List Error	1	0	0	1	1	×	×	×	
	No Defect Spare Location	1	0	1	0	1	×	×	×	
	Seek Error	1	1	0	0	1	×	×	×	

640KB インターフェース・モードにするためには、ディップ・スイッチ(SW₃₋₁/SW₃₋₂)で指定して、再立ち上げる必要があります。通常は 1MB インターフェース・モードで使われます。この場合は、DMA や割り込み等も 1MB 専用インターフェースと同じものが使用されます。

1MB インター・フェース・モードで 640KB のフロッピー・ディスクをアクセスするときは、デバイス・タイプに 70H~73H ではなく 10H~13H を使用します。

◆ 320KB インターフェース

320KB フロッピー・インターフェースは、PC9801/E/F/M でのみ使用可能です。デバイス・タイプ(DA)には 50H~53H を使用します。

1MB/640KB 両用ドライブでは、320KB のディスクを読むモードがあります。これは、320KB(2D)と 640KB(2DD)のメディアの記憶密度、記憶方式が同じでトラックの密度だけが違うため、トラックの移動ステップを 2 倍にすることで、640KB モードで 2D を読み出すしくみです。しかし、ヘッドの幅が 2D と 2DD では違うために、書き込みは保証されません。

◆ 固定ディスク BIOS コマンド

● エラー・リトライ処理

エラー・リトライは、ハードウェアで 8 回まで行える設定にできます。

● 相対アドレス

セクタの指定方法には、シリンダ・ヘッド・セクタを別々に指定する「絶対セクタ・アドレス」と、0

~2097152(21 ビット)のシーケンシャル番号で指定する「相対アドレス」の 2 種類でアクセスできます。

絶対セクタ・アドレスでは、CX(シリンダ)、DH(ヘッド)、DL(セクタ)で指定し、相対セクタ・アドレスでは、DH:DL:CH:CL(SASI の場合使用するのは 21 ビット)で指定します。

● IDE・HDD の違い

IDE ではリトラクトはサポートしていません。コマンド実行時は何もせずに、またエラーも出ません。トラック・フォーマット・コマンドを実行しても物理フォーマットはされず、0 シリンダ、0 トラックの先頭 16K バイトに E5H を書き込むだけです。

物理セクタは 512 バイトに固定されているので、セクタ長を 256 バイトに指定した場合はソフトウェア・エミュレートされます。

98NOTE 等では、一定時間アクセスがないときに、モータ OFF にできます。次にアクセスした場合は自動的にモータを ON にしますが、回転が安定するまで通常 4~5 秒かかるために、最初のアクセスは遅くなります。

IDE の HDD を持った機種を判定するには、図 4-17 に示すシステム共通域のビットを確認します。

◆ RAM ドライブ BIOS

RAM ドライブは、1MB/640KB 両用タイプのフロッピー・ディスク・ドライブと BIOS レベルで互換性があるために、ソフトウェアからは同等に扱うことができます。実際に利用する場合は、1MB/640KB 両用タイプ用の BIOS を使用します。

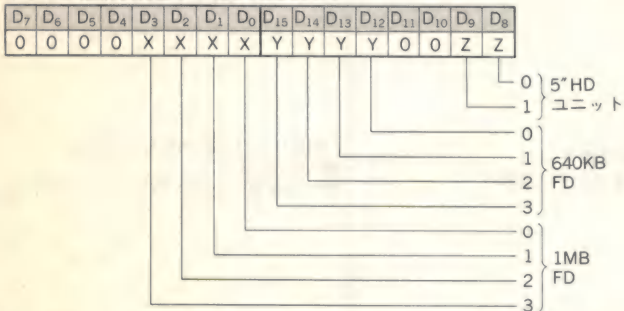
〈図 4-17〉 システム共通域のビットの確認

		000 : 480H	0000 : 480H ビット 7
NS	固定ディスク内蔵モデル	≠00	=1
NS/E	FD モデル	≠00	=0
NC	FD モデル+従来固定ディスク	≠00	=0
他機種	固定ディスク内蔵モデル	=00	=1
	FD モデル	=00	=0
	FD モデル+従来固定ディスク	=00	=0

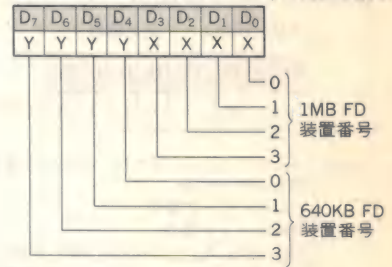
システム共通域 0000 : 480H ビット 7=1 かつ, 0000 : 457H ≠00

〈図 4-18〉 RAM ドライブ装置の確認

ドライブ接続状況ビット (DISK_EQUIP)



RAM ドライブ接続状況ビット (RDISK_EQUIP)



● RAM ドライブ装置の確認方法

RAM ドライブ装置の確認は、システム共通領域 (図 4-18) のドライブ認識ビット (DISK_EQUIP)、RAM ドライブ接続状況認識ビット (RDISK_EQUIP) を参照します。

DISK_EQUIP は、実際の FDD だけでなく、RAM ドライブの搭載でもビットが立つので、RDISK_EQUIP のビットと照らし合わせて、ともに

同じ装置番号にビットが立っていれば、RAM ドライブになります。

以上の情報は、装置の有無を示すもので、RAM ドライブのフォーマット状態は、BIOS を使用して調べる必要があります。

ハード・ディスク BIOS

割り込み番号 INT	機能番号 AH	入力パラメータ	戻り値	機能																																								
1BH	01H	<p>AH=BIOS コマンド識別コード</p> <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>MT</td><td>MF</td><td>r</td><td>Seek</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td colspan="8">(FDD)</td></tr><tr><td>0</td><td>e</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td colspan="8">(HDD)</td></tr></table> <p>AL=デバイス・タイプ・ユニット番号 BX=データ長 CL(CX)=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 CH=セクタ長*1 ES:BP=データ・バッファ領域の先頭アドレス</p>	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	MT	MF	r	Seek	0	0	0	1	(FDD)								0	e	r	0	0	0	0	1	(HDD)								<p>CF=終了条件 AH=ステータス情報</p>	<p>「ペリファイ」 種別=1MB/640KB/2MODE/320KB/ SASI/SCSI 指定したデバイス・タイプ・ユニット の指定された場所のデータを読み出すが、 データのメモリへの転送は行われない。</p>
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
MT	MF	r	Seek	0	0	0	1																																					
(FDD)																																												
0	e	r	0	0	0	0	1																																					
(HDD)																																												

* 1 HDD 時は、シリンダ番号は「CX レジスタ」を使用し、セクタ長 (CH レジスタ) は使用しない

* 2 HDD 時のセンス情報

* 3 ES:BP=フォーマットするセクタの「シリンダ番号」、「ヘッド番号」、「セクタ番号」、「セクタ長」を 1 トラック当たりのセクタ数分指定する。BX=ES:BP で指定したデータ列の長さ

INT	AH	入力パラメータ	戻り値	機能																																			
1BH	O2H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>MF</td><td>r</td><td>Seek</td><td>0</td><td>0</td><td>1</td><td>0</td></tr></table> (FDD) AL=デバイス・タイプ・ユニット番号 BX=データ長 CL=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 CH=セクタ長 ES:BP=データ・バッファ領域の先頭アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	MF	r	Seek	0	0	1	0	CF=終了条件 AH=ステータス情報	「診断のための読み出し(フロッピー・ディスク)」 種別=1MB/2MODE インデックス・マークの直後から読み取りを開始し、IDのエラー、データ部のエラーがあっても読み取りを続けることを除いて「データの読み出し」と同じ機能を持つ。																			
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																															
	0	MF	r	Seek	0	0	1	0																															
	O2H	AH=BISO コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>e</td><td>r</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr></table> (HDD) AL=デバイス・タイプ・ユニット番号 BX=データ長 CL=シリンダ番号 DH=ヘッド番号 DL=ディスク・アドレス(セクタ番号) ES:BP=データ・バッファ領域の先頭アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	e	r	0	0	0	1	0	CF=終了条件 AH=ステータス情報	「診断のための読み出し(SCSI)」 種別=SCSI データを書いた後、ECC チェックする。																			
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																															
0	e	r	0	0	0	1	0																																
O3H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr></table> (FDD) (HDD) AL=デバイス・タイプ・ユニット番号	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	CF=終了条件 AH=ステータス情報	「初期化」 種別=1MB/640KB/2MODE/320KB/SASI/SCSI ディスク装置、コントローラの初期化を行う。												
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																
0	0	0	0	0	0	1	1																																
0	0	0	0	0	0	1	1																																
O4H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>n</td><td>0</td><td>r</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr></table> (FDD) AL=デバイス・タイプ・ユニット番号	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	n	0	r	0	0	1	0	0	CF=終了条件 AH=ステータス情報 * 2	「センス(フロッピー・ディスク)」 種別=1MB/640KB/2MODE/320KB 指定したデバイスの状態を調べる。																				
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																
n	0	r	0	0	1	0	0																																
O4H	AH=BIOS コマンド識別コード O4H/44H/84H AL=デバイス・タイプ・ユニット番号 <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>d₇</td><td>d₆</td><td>d₅</td><td>d₄</td><td></td><td></td><td></td><td></td></tr></table> <div>接続されているデバイスの容量を通知する</div> <table><tr><td>5MB</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>10MB</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>20MB</td><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>40MB</td><td>0</td><td>1</td><td>0</td><td>0</td></tr></table> <div>d₇ d₆ d₅ d₄ については「ステータス一覧」を参照</div> <div>AH=44H の場合(SCSI) BX=1(ソフト・セクタ) =2(ハード・セクタ) AH=84H の場合(SASI/SCSI) BX=セクタ長 CX=シリンダ番号 DH=ヘッド番号 DL=セクタ数</div>	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	d ₇	d ₆	d ₅	d ₄					5MB	0	0	0	0	10MB	0	0	0	1	20MB	0	0	1	1	40MB	0	1	0	0	CF=終了条件 AH=ステータス情報 AH=00H の場合(SASI)	「センス(ハード・ディスク)」 種別=SASI/SCSI デバイスの状態、諸元を読む。
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																
d ₇	d ₆	d ₅	d ₄																																				
5MB	0	0	0	0																																			
10MB	0	0	0	1																																			
20MB	0	0	1	1																																			
40MB	0	1	0	0																																			

INT	AH	入力パラメータ	戻り値	機能																																								
1BH	05H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>MT</td><td>MF</td><td>r</td><td>Seek</td><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td colspan="8">(FDD)</td></tr><tr><td>0</td><td>e</td><td>r</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td colspan="8">(HDD)</td></tr></table> AL=デバイス・タイプ・ユニット番号 BX=データ長 CL(CX)=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 CH=セクタ長*1 ES:BP=データ・バッファ領域の先頭アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	MT	MF	r	Seek	0	1	0	1	(FDD)								0	e	r	0	0	1	0	1	(HDD)								CF=終了条件 AH=ステータス情報	「データの書き込み」 種別=1MB/640KB/2MODE/320K
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																				
	MT	MF	r	Seek	0	1	0	1																																				
	(FDD)																																											
0	e	r	0	0	1	0	1																																					
(HDD)																																												
06H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>MT</td><td>MF</td><td>r</td><td>Seek</td><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td colspan="8">(FDD)</td></tr><tr><td>0</td><td>e</td><td>r</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td colspan="8">(HDD)</td></tr></table> AL=デバイス・タイプ・ユニット番号 BX=データ長 CL(CX)=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 CH=セクタ長*1 ES:BP=データ・バッファ領域の先頭アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	MT	MF	r	Seek	0	1	1	0	(FDD)								0	e	r	0	0	1	1	0	(HDD)								CF=終了条件 AH=ステータス情報	「データの読み出し」 種別=1MB/640KB/2MODE/320KB/ SASI/SCSI 指定したデバイス・タイプ・ユニットから、指定された場所のデータを、指定されたアドレスへ読み出す。	
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
MT	MF	r	Seek	0	1	1	0																																					
(FDD)																																												
0	e	r	0	0	1	1	0																																					
(HDD)																																												
07H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>0</td><td>r</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td colspan="8">(FDD)</td></tr><tr><td>0</td><td>0</td><td>r</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td colspan="8">(HDD)</td></tr></table> AL=デバイス・タイプ・ユニット番号	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	r	0	0	1	1	1	(FDD)								0	0	r	0	0	1	1	1	(HDD)								CF=終了条件 AH=ステータス情報	「シリンダ0へのシーク」 種別=1MB/640KB/2MODE/SASI/ SCSI 指定したデバイス・タイプ・ユニットに対して、0シリンダへシークさせる。	
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
0	0	r	0	0	1	1	1																																					
(FDD)																																												
0	0	r	0	0	1	1	1																																					
(HDD)																																												
09H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>MT</td><td>MF</td><td>r</td><td>Seek</td><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td colspan="8">(FDD)</td></tr></table> AL=デバイス・タイプ・ユニット番号 BX=データ長 CL=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 CH=セクタ長 ES:BP=データ・バッファ領域の先頭アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	MT	MF	r	Seek	1	0	0	1	(FDD)								CF=終了条件 AH=ステータス情報	「デリーテッド・データの書き込み (フロッピー・ディスク)」 種別=1MB/2MODE セクタのデータ・フィールドの Data Address Mark の代わりに Delated Data Address Mark を書き込むことを除いては「データの書き込み」と同じ機能を持つ。																	
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
MT	MF	r	Seek	1	0	0	1																																					
(FDD)																																												
09H	AH=BIOS コマンド識別コード 09H AL=デバイス・タイプ・ユニット番号 BX=データの長さ ES:BP=DEFECT LIST の先頭アドレス (図 4-19 参照)	CF=終了条件 AH=ステータス情報	「不良セクタの代替(SCSI)」 不良セクタを代替する。																																									

INT	AH	入力パラメータ	戻り値	機能																
1BH	OAH	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>MF</td><td>r</td><td>Seek</td><td>1</td><td>0</td><td>1</td><td>0</td></tr></table> (FDD) AL=デバイス・タイプ・ユニット番号 CL=シリンダ番号(AHのSEEKビットが1のときに意味を持つ) DH=ヘッド番号	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	MF	r	Seek	1	0	1	0	CF=終了条件 AH=ステータス情報 CH=セクタ長 CL=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 機能=「IDの読み出し(フロッピー・ディスク)」 種別=1MB/640KB/2MODE 指定したデバイス・タイプ・ユニットの指定されたトラックで、最初に正常に読み出せるIDをID情報に格納する。	
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀												
	0	MF	r	Seek	1	0	1	0												
	OAH	AH=BIOS コマンド識別コード OAH AL=デバイス・タイプ・ユニット番号 BH=セクタ長 1=256 バイト 2=512 バイト 3=1024 バイト	CF=終了条件 AH=ステータス情報	「セクタ長指定(SCSI)」 セクタ長を指定する。																
OCH	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>MT</td><td>MF</td><td>r</td><td>Seek</td><td>1</td><td>1</td><td>0</td><td>0</td></tr></table> (FDD) AL=デバイス・タイプ・ユニット番号 BX=データ長 CL=シリンダ番号 DH=ヘッド番号 DL=セクタ番号 CH=セクタ長 ES:BP=データ・バッファ領域の先頭アドレス	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	MT	MF	r	Seek	1	1	0	0	CF=終了条件 AH=ステータス情報	「デリーテッド・データの読み出し(フロッピー・ディスク)」 種別=1MB/2MODE セクタのデータ・フィールドの、「Data Address Mark」の代わりに「Deleted Data Address Mark」を扱うことを除いては「データの読み出し」と同じ機能を持つ。	
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀													
MT	MF	r	Seek	1	1	0	0													
OCH	AH=BIOS コマンド識別コード OCH/8CH OCH=READ DEFECT DATA 8CH=READ DEFECT NUMBER AL=デバイス・タイプ・ユニット番号 BX=DEFECT LISTの長さ (READ DEFECT DATA) ES:BP=DEFECT LISTを格納する先頭アドレス (READ DEFECT DATA)	CF=終了条件 AH=ステータス情報 CX=代替されている不良セクタの数 (READ DEFECT NUMBER)	「代替情報取得(SCSI)」 代替を行っているセクタの情報/数を返す。																	
	ODH	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>MF</td><td>r</td><td>Seek</td><td>-1</td><td>1</td><td>0</td><td>1</td></tr></table> (FDD) AL=デバイス・タイプ・ユニット番号 BX=データ長*3 CH=セクタ番号 CL=シリンダ番号 DH=ヘッド番号 DL=データ部への書き込みデータ・パターン ES:BP=データ・バッファ領域の先頭アドレス*3	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	MF	r	Seek	-1	1	0	1	CF=終了条件 AH=ステータス情報	「フォーマット(フロッピー・ディスク)」 種別=1MB/640KB/2MODE/320KB 指定した1トラックを任意の形式でフォーマットする。
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀													
0	MF	r	Seek	-1	1	0	1													

INT	AH	入力パラメータ	戻り値	機 能																																
1BH	ODH	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>d</td><td>e</td><td>r</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr></table> (HDD) d=0 トラック単位 (SASI のみ) d=1 ドライブ単位 <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>d</td><td>x</td><td>r</td><td>x</td><td>1</td><td>1</td><td>0</td><td>1</td></tr></table> *D ₇ =フォーマット単位の指定 0: トラック単位 1: デバイス単位 AL=デバイス・タイプ・ユニット番号 BH=インタリーブ・ファクタ (1~16) CX=シリンダ番号 DH=ヘッド番号 DL=0 ※ドライブ単位でフォーマットする場合は、CX=0, DH=0 とする。	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	d	e	r	0	1	1	0	1	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	d	x	r	x	1	1	0	1	CF=終了条件 AH=ステータス情報	「フォーマット (ハード・ディスク)」 種別=SASI/SCSI 指定したトラック (SASI のみ)/ドライブをフォーマットする。
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																												
	d	e	r	0	1	1	0	1																												
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																													
d	x	r	x	1	1	0	1																													
			<div>〈図 4-19〉 不良セクタの代替</div> <div><div>ES/BP→</div><div><table><tr><td>0000H</td><td>(ワード)</td></tr><tr><td>不良情報の長さ</td><td>= n × 4 (ワード)</td></tr><tr><td>不良セクタの論理アドレス 1</td><td rowspan="3">} 4 バイト (Low バイト→ High バイトの順)</td></tr><tr><td>⋮</td></tr><tr><td>不良セクタの論理アドレス n</td></tr></table></div></div>		0000H	(ワード)	不良情報の長さ	= n × 4 (ワード)	不良セクタの論理アドレス 1	} 4 バイト (Low バイト→ High バイトの順)	⋮	不良セクタの論理アドレス n																								
0000H	(ワード)																																			
不良情報の長さ	= n × 4 (ワード)																																			
不良セクタの論理アドレス 1	} 4 バイト (Low バイト→ High バイトの順)																																			
⋮																																				
不良セクタの論理アドレス n																																				
	OFH	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>0</td><td>r</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table> (HDD) AL=デバイス・タイプ・ユニット番号	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	r	0	1	1	1	1	CF=終了条件 AH=ステータス情報	機能=「リトラクト」 種別=SASI/SCSI 未使用シリンダへ、ヘッドを退去させる。																
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																													
0	0	r	0	1	1	1	1																													
	10H	AH=BIOS コマンド識別コード <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>0</td><td>r</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table> (FDD) AL=デバイス・タイプ・ユニット番号 CL=シリンダ番号	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	r	1	0	0	0	0	CF=終了条件 AH=ステータス情報	「シーク」 種別=1MB/640KB/2MODE/320KB 指定したデバイス・タイプ・ユニットに対して、指定されたシリンダまでシークする。																
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																													
0	0	r	1	0	0	0	0																													

1MB/640KB 両用 フロッピー・ディスク専用 BIOS

割り込み番号 INT	機能番号 AH	入力パラメータ	戻り値	機能																
1BH	OE H	<p>AH=BIOS コマンド識別コード 0EH/8EH AL=各ユニットの動作モード</p> <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td></td><td></td><td></td><td></td></tr></table> <p>D₀=Unit #0 D₁=Unit #1 D₂=Unit #2 D₃=Unit #3 AH=0EH のとき、上記各 Unit の bit は "0": 片面モード, "1": 両面モード AH=8EH のとき、上記各 Unit の bit は "0": 48tpi モード, "1": 96tpi モード 注: 各ビットの初期状態はすべて "1" である。これらの指定は、 DA=1×H のコマンドに対して 有効となる。</p>	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	0	0	0	1					<p>CF=終了条件 CF=0, AH=00H 正常終了 CF=1, AH=40H 異常終了</p>	<p>「動作モードの設定」 種別=2MODE (1MB モード) 両用ドライブを 1MB モードで使用する 際に、1MB/640KB のアクセス・モード を設定する。</p>
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀													
0	0	0	1																	

INT	AH	入力パラメータ	戻り値	機能																																
1BH	83H	AH=BIOS コマンド識別コード 83H AL=デバイス・タイプ・ユニット番号 (PC9801/E/F/M では、7×H のときに正常終了となるので F×H を使用するほうが望ましい)	CF=終了条件 CF=0, AH=OOH 正常終了 CF=1, AH=40H 異常終了	「初期化」 種別=2MODE(640KB モード) 両用ドライブを 640KB インターフェース・モード時に、AI を検出するように初期化する。																																
	83H	AH=BIOS コマンド識別コード 83H AL=デバイス・タイプ・ユニット番号	CF=終了条件 CF=0, AH=OOH 正常終了 CF=1, AH=40H 異常終了 (外付けユニットにこのコマンドを実行しても、つねに正常終了が返る)	「モータ停止モードの設定」 種別=2MODE(1MB モード) 1MBFD のモータを自動的に ON/OFF するモードに設定する。																																
	84H	AH=BIOS コマンド識別コード 84H AL=デバイス・タイプ・ユニット番号	CF=終了条件 AH=ステータス情報	「センス」 種別=2MODE 指定したデバイスの状態を調べる。																																
	84H	<div>IMB インターフェース・モード時</div> <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>0</td></tr></table> <div>・1MB フロッピー・ディスク BIOS 「センス AH=04H」と同じ</div> <div>640KB インターフェース・モード時</div> <table><tr><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table> <div>・640KB フロッピー・ディスク BIOS 「センス AH=04H」と同じ</div> <div>D₀="0": 片面媒体 "1": 両面媒体 (1MB/640KB 両用ドライブでは常に両面) D₃="0": IMB ドライブ "1": 1MB/610KB 両用ドライブ D₀="0": 片面モード "1": 両面モード D₁="0": AI あり (1MB ドライブのときは必ず 0) "1": AI なし D₂="0": 48tpi (40 シリンダ) モード "1": 96tpi (80 シリンダ) モード 640KB タイプのときのみ有効。 1MB タイプでは必ず 0 となる D₃="0": 640KB ドライブ "1": 1MB/640KB 両用ドライブ 注: エラー発生時は、D₃~D₀ のうち両用タイプ・ビット (D₃) のみ有効</div>			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀							0	0	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀								
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																												
						0	0																													
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																													
9EH	AH=BIOS コマンド識別コード 9EH AL=デバイス・タイプ・ユニット番号 (1CH~1DH/内蔵ドライブのみ)	CF=終了条件 CF=0, AH=OOH 正常終了 CF=1, AH=40H 異常終了	「2D モードに切り替える」																																	
9EH	AH=BIOS コマンド識別コード 9EH AL=デバイス・タイプ・ユニット番号 (1EH~1FH/内蔵ドライブのみ)	CF=終了条件 CF=0, AH=OOH 正常終了 CF=1, AH=40H 異常終了	「2DD モードに切り替える」																																	

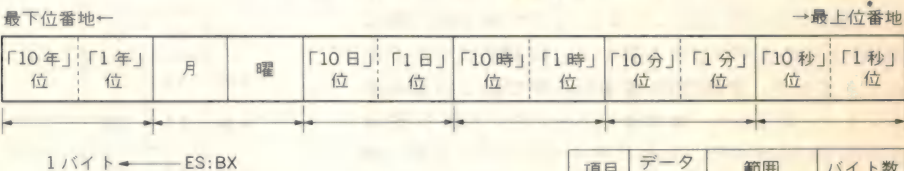
タイマ BIOS

タイマ BIOS は、カレンダー時計の設定・読み出し、インターバル・タイマの設定等を行います。

割り込み番号 INT	機能番号 AH	入力パラメータ	戻り値	機能
1CH	OOH	ES: BX=読み出した日付けを格納するデータ・バッファ (6 バイト) のアドレス	データ・バッファに格納される	日付け・時刻の読み出し 現在の年、月、曜日、日、時、分、秒をカレンダー時計 LSI から読み出して、データ・バッファに格納する (図 4-20 参照)。 カレンダー時計 LSI への直接のアクセスは避けて、BIOS を使用するべきである。

INT	AH	入力パラメータ	戻り値	機能
1CH	01H	ES: BX=読み出した日付け, 時刻を格納されているデータ・バッファ(6バイト)のアドレス.	なし	「日付け・時刻の設定」 データ・バッファに格納されている, 年, 月, 曜日, 日, 時, 分, 秒をカレンダー時計 LSI へ設定する. データ・バッファ形式は「日付け・時刻の読み出しと同じ」
1CH	02H	CX=インターバル・タイマ値 n (単位は 10ms) $1 \leq n \leq 65536$ ($n=65536$ は $n=0$ と設定する) ES: BX=ユーザのタイマ割り込み処理ルーチンのアドレス.	なし	「インターバル・タイマの設定(シングル・イベント)」 インターバル・タイマ値を設定し, 起動する. 設定値まで時間が経過すると割り込みが発生し, ユーザのタイマ割り込み処理ルーチンに制御を移す. 1 回の設定で 1 回だけインターバル・タイマが起動される(リスト 4-1 参照).

<図 4-20> 日付け・時刻の読み出し



<リスト 4-1> インターバル・タイマの設定(サンプル・プログラム)

項目	データ形式	範囲	バイト数
年	BCD	00H~99H	1
月	16 進数	01H~0CH	1
曜	16 進数	00H~06H	
日	BCD	01H~31H	1
時	BCD	00H~23H	1
分	BCD	00H~59H	1
秒	BCD	00H~59H	1

注: PC9801/E/F1, 2, 3/M2, 3/U2/VF0, 2, 4/UV2/PC-98XA では, 「年」は不揮発メモリに格納されるために, 自動的に更新されない. また, うるう年もサポートされない.
 「月」は月の大小を自動判別する

```

/*
** インターバルタイマーのテスト
**
** 5 秒間の間に何回メモリアクセスができるか数える.
**
#include <dos.h>
#include <stdio.h>

int endflg = 0;

/*
** インターバルタイマの設定
**
*/
void settimer(unsigned int count,void interrupt (*intprg)())
{
    struct REGPACK reg;

    reg.r_ax = 0x0200; /* インターバルタイマの設定 */
    reg.r_cx = count; /* インターバルタイマ値 nx10ms */
    reg.r_es = FP_SEG(intprg); /* タイマ処理ルーチンのセグメント */
    reg.r_bx = FP_OFF(intprg); /* タイマ処理ルーチンのオフセット */
    intr(0x1c, &reg); /* タイマ BIOS をコール */
}

/*
** タイマー割り込み処理部
**
** 終了のフラグをたてる.
**
*/
void interrupt timer(void)
{
    endflg = 1;
}

void main(void)
{
    unsigned int i, j, dmy=0;

    printf("簡易ベンチマーク (5 秒お待ち下さい) >");
    settimer(500, timer); /* インターバルタイマの設定 500x10ms */
    for (i = 0; i < 60000; i++) {
        for (j = 0; j < 0x1fff; j++) {
            dmy += peek(j << 3, 0);
            if (endflg != 0) break;
        }
        if (endflg != 0) break;
    }
    printf("%u\n", i);
}

```


グラフィック LIO

拡張グラフィック機能(4096色中16色モード、GRGC/EGCを使用した高速描画等)はグラフ LIO によりサポートされます。機種や動作状況によっては動作しないものもあります。

グラフ LIO は N₈₈BASIC で使用するために作られたプログラム・モジュールです。アセンブラや C 言語等で使用する場合は、必ずしも使いやすいとはいいいくのですが、グラフィック BIOS や GDC 直接操作よりは、随分と良くなります。

◆ グラフ LIO の使用方法

グラフ LIO (以降、LIO と略す) は、BIOS のように割り込みによる呼び出しではなく、PC98 本体内蔵の ROM 上にあるプログラム群です。17 種のコマンドからできており、F9900H 番地から呼び出しのためのエントリ・ポイント・オフセット・アドレスが 17 個分書かれていますので、これに従って呼び出します。使用する場合は、割り込みベクタ(ソフトウェア割り込み)にこのアドレスを書き込み、INT 命令で呼び出すのが普通になっています。N₈₈BASIC では、AOH ~ AFH、CEH の割り込みを使用しますが、自由に變更できます(図 4-21)。

LIO を使用する前に、ワーク・エリア(UCW)と、スタック・エリアを確保しなくてはなりません。

UCW は 1400H バイト(GCOPY コマンドを使用しない場合は 1200H バイト)分必要です。セグメント・レジスタ「DS」でこのアドレスを指定します。オフセットは「0000H~13FFH」が UCW として使用されます。

スタック・エリアは LIO が専用に 128 バイト使用します。レジスタ「SS:SP」で指定します。

必要に応じて、LIO のエント・ポイントを割り込みベクタにセットします。以下の説明では、N₈₈BASIC と同様に、AOH~AFH、CEH の割り込みベクタを使用することを前提とします。

◆ グラフ LIO のコマンド

グラフ LIO の入出力パラメーター一覧を図 4-22 に示します。

C 言語によるグラフィック操作

グラフィック画面を扱うには、GDC 直接操作やグラフィック BIOS、LIO 等があります。GDC や BIOS の操作は、比較的難しく機能も低くなっています。LIO の操作は、比較的優しく機能も多くなっています

〈図 4-21〉 グラフ LIO の使用方法

エントリ数			
F9900H	+0	11	
	+4	A0	00
	+8	A1	00
	+12	A2	00
	+16	A3	00
	+20	A4	00
	+24	A5	00
	+28	A6	00
	+32	A7	00
	+36	A8	00
	+40	A9	00
	+44	AA	00
	+48	AB	00
	+52	AC	00
	+56	AD	00
	+60	AE	00
	+64	AF	00
	+68	CE	00
	+70	00	00
	+72	ID 情報	
		エントリ・ポイントの オフセット・アドレス	

注 1: 各コマンドのエントリ・ポイントのセグメント・ベースは、F9900H (セグメント・レジスタへの格納値は F990H)

注 2: N₈₈ BASIC システムでの割り込みベクタ番号は AO ~ AF、CE を使用

が、使用するにはアセンブラ・レベルの知識が多く必要になります。

最近では、コンパイラ言語が、安く手軽に使えるようになってきました。CPU の速度も高速になってきましたので、コンパイラ言語だけでプログラムを作成しても速度的に十分なことが多く、使用する頻度も上がってきています。

これらのコンパイラ言語には、独自にグラフィック操作命令を持つものも多くあります。そこで、C 言語 (Turbo C++ Ver.1.00) における、グラフィック関連の命令(関数)と、コンパイル時の注意について解説します。

〈図 4-22〉 グラフ LIO の入出力パラメータ

割り込み番	ルーチン名	機 能	入 力 パ ラ メ ー タ	出 力 パ ラ メ ー タ
A0h	GINIT	初期化	なし	AH=00h:正常終了
A1h	GSCREEN	モード設定	BX:パラメータ・リストへのポインタ(図 a)	AH=00h:正常終了, AH=05h:不正呼び出し
A2h	GVIEW	ビューポート指定	BX:パラメータ・リストへのポインタ(図 b)	AH=00h:正常終了, AH=05h:不正呼び出し
A3h	GCOLOR1	背景色指定	BX:パラメータ・リストへのポインタ(図 c)	AH=00h:正常終了
A4h	GCOLOR2	パレット番号と表示色の対応	BX:パラメータ・リストへのポインタ(図 d)	AH=00h:正常終了
A5h	GCLS	描画領域の塗りつぶし	なし	AH=00h:正常終了
A6h	GPSET	点を打つ	BX:パラメータ・リストへのポインタ(図 e) AH=01h:フォア・グラウンド・パレット番号 AH=02h:バック・グラウンド・パレット番号	AH=00h:正常終了
A7h	GLINE	直線, 方形を描く	BX:パラメータ・リストへのポインタ(図 f)	AH=00h:正常終了
A8h	GCIRCLE	円, 楕円を描く	BX:パラメータ・リストへのポインタ(図 g)	AH=00h:正常終了, AH=06h:演算オーバフロー
A9h	GPAINT1	色で塗りつぶし	BX:パラメータ・リストへのポインタ(図 h)	AH=00h:正常終了, AH=05h:不正呼び出し AH=07h:ワーク域不足
AAh	GPAINT2	タイルで塗りつぶし	BX:パラメータ・リストへのポインタ(図 i)	AH=00h:正常終了, AH=05h:不正呼び出し AH=07h:ワーク域不足
ABh	GGET	描画情報の格納	BX:パラメータ・リストへのポインタ(図 j)	AH=00h:正常終了, AH=05h:不正呼び出し
ACH	GPOT1	描画情報の表示	BX:パラメータ・リストへのポインタ(図 k)	AH=00h:正常終了, AH=05h:不正呼び出し
ADh	GPOT2	日本語の描画	BX:パラメータ・リストへのポインタ(図 l)	AH=00h:正常終了, AH=05h:不正呼び出し
AEh	GROLL	描画面面の移動	BX:パラメータ・リストへのポインタ(図 m)	
AFh	GPOINT2	ドットのパレット番号の取得	BX:パラメータ・リストへのポインタ(図 n)	AH=00h:正常終了, AL:パレット番号
CEh	GCOPY	ドット情報の格納	AX:左上点X座標 BX:左上点Y座標 CL:X方向ドット数 CH:Y方向ドット数 DI:バッファのオフセット・アドレス ES:バッファのセグメント・アドレス	AH:不定

(a) GSCREEN のパラメータ・フォーマット

	BX+0	+1	+2	+3	+4
	画面 モード		画面 スイッチ	アクティブ・ ページ	ディスプレイ・ ページ
パラ メータ	画面モード	画面スイッチ	アクティブ・ ページ	ディスプレイ・ ページ	
00h	カラー・モード (640×200ドット)	表示あり	アクティブ・ ページの番号 (0～11)	ディスプレイ・ ページの番号 (0～31)	
01h	モノクロ・モード (640×200ドット)	表示あり 高速書き込み			
02h	モノクロ・モード (640×400ドット)	表示なし			
03h	カラー・モード (640×400ドット)	表示なし 高速書き込み			
FFh	今までのモードを引き継ぐ				

(c) GCOLOR1 のパラメータ・フォーマット

BX+0	+1	+2	+3	+4
未使用	バック・グラウンド・カラー (0~7)	ボーダー・カラー (0~7)	フォア・グラウンド・カラー (0~7)	

(d) GCOLOR2 のパラメータ・フォーマット

BX+0	+2	+3
パレット番号 (0~7)	表示色コード (0~7)	

(e) GPSET のパラメータ・フォーマット

BX+0	+1	+2	+3	+4	+5
X座標		Y座標		パレット番号 (0~7)	

(b) GVIEW のパラメータ・フォーマット

BX+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10
左上X座標 (X ₁)	左上Y座標 (Y ₁)	右下X座標 (X ₂)	右下Y座標 (Y ₂)	領域色	境界色					
					パレット番号 (0~7) 外枠を描かない(FFFh)					
					パレット番号(0~7) 塗りつぶししない(FFFh)					

(f) GLINE のパラメータ・フォーマット

BX +0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+11	+12	+13	+14
始点の X座標(X ₁)		始点の Y座標(Y ₁)		終点の X座標(X ₂)		終点の Y座標(Y ₂)		パレット 番号1 (0~7)	描画指定 0:直線 1:方 2:方 塗りつぶし	ライン・ スタイル スイッチ	パレット 番号2	ライン・スタイル LOW HI		タイトル・ パターン 長

- トランジスタ技術
-
- SPECIAL

(k) GPUT1のパラメータ・フォーマット

BX+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+11	+12	+13	+14
左上点 X座標(X)		左上点 Y座標(Y)		格納域 オフセット・アドレス	格納域 セグメント・アドレス	格納域 長さ(バイト)		描画 モード	カラー スイッチ	フォア・ グラウンド・ カラー	バック・ グラウンド・ カラー			

(0~7)

▶ 描画モード(+10)

指定領域上のパターンと
格納域パターンとの論理演算

00h: T
01h: NOT
02h: OR
03h: AND
04h: XOR

(m) GROLLのパラメータ・フォーマット

BX+0	+1	+2	+3	+4	+5
上下ドット数 (-399~399)	左右ドット数 (-639~639)	フラグ			

▶ フラグ(+4)

00h: 移動後の残り領域を
パレット0にする
01h: 移動後の残り領域を
バック・グラウンド・
カラーにする

(l) GPUT2のパラメータ・フォーマット

BX+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10
左上 X座標(X)	左上 Y座標(Y)	日本語JIS コード	描画 モード	カラー スイッチ	フォア・ グラウンド・ カラー	バック・ グラウンド・ カラー				

(0~7)

▶ 描画モード(+6)

指定領域上のパターンと
格納域パターンとの論理演算
00h: T
01h: NOT
02h: OR
03h: AND
04h: XOR

(n) GPOINT2のパラメータ・フォーマット

BX+0	+1	+2	+3	+4
X座標, (X)	Y座標, (Y)			

面倒な面も多いと思います。開発中はロードして使い
完成したら組み込んで置く等、自分の使い方に応じて
使い分けると良いでしょう。

● ロードして使う方法

プログラムの最初で `initgraph` でロードするグラフィックス・ドライバ(*.BGI)のディレクトリを、次に示す例のように指定します。

```
initgraph(&graphdriver,&graphmode,"a:
¥¥¥turboc¥¥¥bgi¥¥¥");
```

グラフィックス関数を使用する場合、`graphics.lib`をリンクする必要がありますので、次のようにコンパイルします。

tcc ソース・ファイル名 `graphics.lib`

● 組み込んで使う方法

まず、グラフィックス・ドライバ(*.BGI)を `BGI\OBJ.EXE` を使って `OBJ` ファイルに変換します。

例 `bgiobj pc98`

`pc98.bgi` を `pc98.obj` に変換する。拡張子は付けない。

後は、コンパイルのときに次のように作成された `OBJ` ファイルを指定すれば、組み込むことができます。

tcc ソース・ファイル名 `graphics.lib OBJ` ファイル名

■ Turbo C++のグラフィックについて

Turbo C++では、70を超える高機能なグラフィックス関数が用意されており、仮想画面、ライン・スタイル、塗りつぶしパターンの設定等多くの機能がサポートされています。これらの機能はグラフィックス・ドライバによるものです。

グラフィックス・ドライバには、ノーマル(640×400)用の `PC98`、`BGI` とハイレゾ(1120×750)用の `PC98HI`、`BGI` の二つがあります。これらのグラフィックス・ドライバはハードウェアをチェックし、最適なハードウェアを使用することが可能です。

■ グラフィックス・ドライバの使い方

グラフィックス・ドライバは、プログラム起動後ロードして使う方法と実行ファイルに組み込んで使う方法の2通りの方法があります。グラフィックス・ドライバは巨大ですので、ロードする方法は実行ファイルのサイズが小さくなりますし、コンパイル時間も短くなります。しかし、外部にドライバを必要とするのは


```

/*
** グラフィックのテスト
**
** ドライバをロードする場合
**   tcc sample.c graphics.lib
** ドライバを組み込む場合
**   tcc -DDRIVER_LINK sample.c graphics.lib pc98.obj
**
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <graphics.h>

void grapherexit(int errorcode); /* グラフィクス終了 */

int main()
{
    int    gdriver = DETECT; /* 自動検出要求 */
    int    gmode, errcode;

#ifdef DRIVER_LINK /* ドライバを組み込む方法 */
    if ((errcode = registerbgidriver(PC98_driver)) < 0) grapherexit(errcode);
    initgraph(&gdriver, &gmode, "");
#else /* ドライバをロードする方法 */
    initgraph(&gdriver, &gmode, "a:\\turboc\\V\\bg\\");
#endif
    if ((errcode = graphresult()) < 0) grapherexit(errcode);

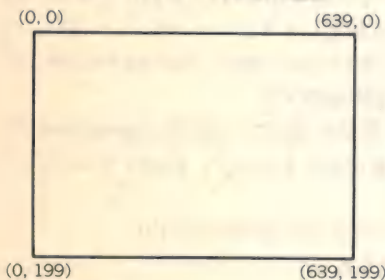
    setcolor(WHITE);
    setfillstyle(CLOSE_DOT_FILL, LIGHTBLUE);
    bar(50, 50, 590, 350);
    setfillstyle(XHATCH_FILL, LIGHTGREEN);
    pieslice(320, 200, 45, 315, 100);

    getch(); /* 何かキーを押したら終了 */
    closegraph();
    return 0;
}

/*
** グラフィックエラー終了
**
void grapherexit(int errorcode)
{
    printf("Graphics error : %d\n", errorcode);
    exit(1);
}

```

〈図 4-23〉 グラフィックの座標



〈図 4-24〉 カラー・コードと色

color	値	実際の色	color	値	実際の色
BLACK	0	黒	DARKGRAY	8	暗い灰色
BLUE	1	青	LIGHTBLUE	9	明るい青
GREEN	2	緑	LIGHTGREEN	10	明るい緑
CYAN	3	水色	LIGHTCYAN	11	明るい水色
RED	4	赤	LIGHTRED	12	明るい赤
MAGENTA	5	紫	LIGHTMAGENTA	13	明るい紫
BROWN	6	暗い黄色	YELLOW	14	黄色
LIGHTGRAY	7	灰色	WHITE	15	白

OBJ ファイルがカレント・ディレクトリにない場合はフルパス指定します。

プログラム中では、initgraphの前にregisterbgidriverでドライバをシステムに登録する必要があります。

例 registerbgidriver(PC98_driver);リンクされているノーマル用ドライバを登録

サンプル・プログラムでは、二つの方法でコンパイル

ルできるように、DRIVER_LINKが定義されているかどうかで条件コンパイルするようになっています。

サンプル・プログラム SAMPLE.Cをリスト 4-2に示します。

ロードして使う場合には、そのままコンパイルします。

tcc sample.c graphics.lib

組み込んで使う場合には、-D オプションを使用し

〈図 4-25〉 エラー・コード

エラー・コード	値	意味
grOK	0	エラーなし
grNoInitGraph	-1	(BGI) グラフィックスがインストールされていない
grNotDetected	-2	グラフィックス・ハードウェアが検出不能である
grFileNotFound	-3	デバイス・ドライバ・ファイルが見つからない
grInvalidDriver	-4	不正なデバイス・ドライバ・ファイル
grNoLoadMem	-5	ドライバをロードするメモリが不足
grNoScanMem	-6	塗りつぶしスキャンでメモリが不足
grNoFloodMem	-7	領域塗りつぶしでメモリが不足
grFontNotFound	-8	フォント・ファイルが見つからない
grNoFontMem	-9	フォントをロードするメモリが不足
grInvalidMode	-10	選択したドライバに対する不正なグラフィックス・モード
grError	-11	グラフィックス・エラー
grIOerror	-12	グラフィックス I/O エラー
grInvalidFont	-13	不正なフォント・ファイル
grInvalidFontNum	-14	不正なフォント番号
grInvalidVersion	-18	ファイルのバージョンが不正

て DRIVER_LINK を定義してコンパイルします。

```
gcc -DDRIVER_LINK sample.c graphics.  
lib pc98.obj
```

■ グラフィック関数

70 以上もあるグラフィック関数すべてを説明するわけにはいきませんので、グラフィックの機能を使用するのに最低限必要と思われるものを図 4-26 にまとめました。

● 座標

座標は画面左上を (0, 0) として x 座標は左から右に向かって増加し、y 座標は上から下に向かって増加します。グラフィックの座標を図 4-23 に示します。

● 色

カラー・コードと色の関係は図 4-24 のとおりです。98 のカラー・コードと異なるのでシンボルを使用しない場合には注意が必要です。8 色モード時に 8~15 を指定した場合、0~7 に変換されます。

エラー・コード一覧を図 4-25 に、また Turbo C++ のグラフィック関連関数のリファレンスを図 4-26 に示します。

●参考文献●

- (1) PC98 シリーズテクニカルデータブック HARDWARE 編, (株)アスキー, 1993 年 10 月 25 日。
- (2) PC98 シリーズテクニカルデータブック BIOS 編, (株)アスキー, 1993 年 4 月 15 日。
- (3) EPSON PC システム・ガイド, (株)クリエイイト・クルーズ, 平成 5 年 3 月 31 日。
- (4) 浅野泰之, 壁谷正洋, 金磯善博, 栗野雅彦; PC-9801 システム解析 (下), (株)アスキー, 1983 年 12 月 1 日。
- (5) トランジスタ技術スペシャル No.3, CQ 出版(株), 昭和 62 年 5 月 1 日。
- (6) 海原系; PC のハードウェアを理解する, 別冊インターフェース BootStrap Project-2 No.4, CQ 出版(株), 1993 年 7 月 1 日。
- (7) NEC 電子デバイス アプリケーション・ノート μ PD71037/ μ PD72020DMA コントローラ, 日本電気(株)。
- (8) μ PD7220GDC ユーザーズ・マニュアル, 日本電気(株), December 1987。
- (9) NEC 電子デバイス ユーザーズ・マニュアル μ PD71059/ μ PD71037/ μ PD71055/ μ PD72020/ μ PD7220A/ μ PD71054, 日本電気(株)。

好評発売中 /

デジタル・オーディオの本です

トランジスタ技術

SPECIAL No.21

B 5 判 160ページ

定価1,540円(税込)送料310円

CQ出版社

最もポピュラーな最新技術を理解しよう

特集 デジタル・オーディオ技術の基礎と応用

Contents

デジタル・オーディオの基礎/デジタル・フィルタの基礎/デジタル・オーディオ・インターフェースの基礎/ $\Delta\Sigma$ 変調方式 D-Aコンバータの基礎/マルチ・エフェクタとサラウンド・プロセッサの製作/DAIボードの製作/デジタル・プリアンプの製作/LSIキットを使ったCDシステム/光磁気ディスク・オーディオ・ファイル装置

● グラフィックス・システムを初期化する

関数/引き数	
initgraph(graphdriver, graphmode, pathdriver);	
解 説	
グラフィックス・ドライバをロードまたは登録済みのドライバを有効にしてグラフィック・システムを初期化する。	
graphdriver	値 意 味
DETECT	0 自動検出を要求する
PC98	1 ノーマル・モードで使用する
PC98HI	2 ハイレゾ・モードで使用する
graphmode	値 意 味
PC98C8	0 8色モードで使用する
PC98C16	1 16色モードで使用する
pathdriver	グラフィックス・ドライバをロードするとき に検索するディレクトリ名で、存在しなかつたときはカレント・ディレクトリを検索する。 graphdriver に DETECT を指定した場合、ハードウェアを自動検出して動作する。エラーが発生しなかった場合、以下の値が代入される。 (graphdriver, graphmode は INT へのポインタ)
graphdriver	値 意 味
PC98	1 ノーマル・モード
PC98HI	2 ハイレゾ・モード
graphmode	値 意 味
PC98C8	0 8色モード
PC98C16	1 16色モード

● グラフィックス・システムの使用を終了する

関数/引き数	
closegraph();	
解 説	
グラフィックス・システムが割り当てたすべてのメモリを解放し、画面モードを initgraph が呼ばれる前の状態にする。	

● ユーザがリンクしたグラフィックス・ドライバをシステムに登録する

関数/引き数	
registerbgidriver(driver);	
解 説	
実行ファイルにグラフィックス・ドライバを組み込んだ場合 initgraph の前にこの関数でシステムに登録する必要がある。エラーが発生した場合、負のグラフィックス・エラー・コードを返す。エラー・コードは図 4-25 参照。	
driver	リンクされているドライバ
PC98-driver	PC98.BGI

PC98HI-driver PC98HI.BGI

● 最後に発生したグラフィックス操作に対するエラーのエラー・コードを返す

関数/引き数	
graphresult();	
解 説	
エラー・コードは図 4-25 参照。	

● グラフィックス画面を消去する

関数/引き数	
cleardevice();	

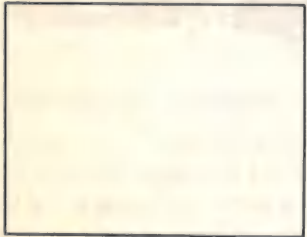
● 描画色を設定する

関数/引き数	
setcolor(color);	
解 説	
color は図 4-24 を参照	

● 塗りつぶしパターンと色を設定する

関数/引き数	
setfillstyle(pattern, color);	
解 説	
塗りつぶしの色は setcolor では変更できない。	
▶ pattern	
EMPTY_FILL	0 背景色で塗る
SOLID_FILL	1 ベタ塗り
LINE_FILL	2 横線で塗る
LTSLASH_FILL	3 /// で塗る
SLASH_FILL	4 /// (太) で塗る
BKSLASH_FILL	5 \\\ (太) で塗る
LTBKSLASH_FILL	6 \\\ で塗る
HATCH_FILL	7 +++ で塗る
XHATCH_FILL	8 xxx で塗る
INTERLEAVE_FILL	9 インターリーブ線で塗る
WIDE_DOT_FILL	10 点(間隔広)で塗る
CLOSE_DOT_FILL	11 点(間隔狭)で塗る
color は図 4-24 を参照。	

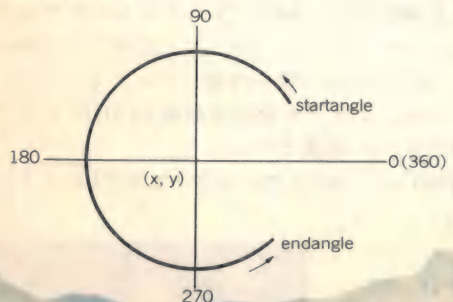
● 長方形を描く

関数／引き数
<code>rectangle(left,top,right,bottom);</code>
解 説
図のような、座標の長方形を描く。 (left, top) (right, top)  (left, bottom) (right, bottom)

● 長方形(塗りつぶし)を描く

関数／引き数
<code>bar(left,top,right,bottom);</code>
解 説
パラメータについては <code>rectangle</code> を参照

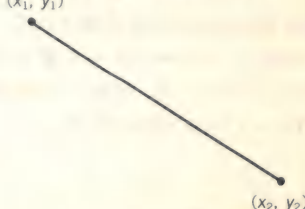
● 円弧を描く

関数／引き数
<code>arc(x,y,startangle,endangle,radius);</code>
解 説
中心座標(x,y), 半径ドット数radius, 描画開始角度startangle, 終了角度endangle, 反時計回りに円弧を描く。startangle, endangle は、時計の3時の位置が0, 12時の位置が90で, startangleはendangleより小さくなるように設定する。 

● 円弧を描きその中を塗りつぶす

関数／引き数
<code>pieslice(x,y,startangle,endangle,radius);</code>
解 説
パラメータについては <code>arc</code> を参照

● 2点間を結ぶ直線を描く

関数／引き数
<code>line(x1,y1,x2,y2);</code>
解 説
2点(x1,y1)と(x2,y2)を結ぶ直線を描く。 

● 指定座標に点を表示する

関数／引き数
<code>putpixel(x,y,color);</code>
解 説
(x,y)に color の色の点を表示する。color は図 4-24 参照

● 指定座標に文字列をグラフィック画面に描く

関数／引き数
<code>outtextxy(x,y,text);</code>
解 説
x,y 書き始め座標, text 描画する文字列

5

AD78090-4を使った 拡張基板の製作



A-D コンバータの製作

最近の A-D コンバータは、多機能、高性能になり、ワンチップで、マルチプレクサ、サンプル&ホールド、その他の制御信号生成部分等が内蔵されて、あとは必要に応じた、CPU インターフェース部とアナログ部 (アンチエリアス・フィルタ、バッファ等) を外付けするだけで A-D コンバータが作れます。

◆ 目標

単電源の OP アンプを使用する機会が多いので、測定範囲としては、0~12 V を中心に、10 kHz 程度の波形を観測することを目標にしました。サンプリング周期は 10 μ s 程度、分解能は 8~12 ビット程度を目標とします。

PC9801 等に接続するには、パラレル出力タイプの A-D コンバータ (以下 ADC と略す) のほうがインターフェースが簡単に作れて便利です。反面、ノイズだらけのコンピュータ内部の拡張基板上に、ADC やアナログ関係の回路を載せなくてはならないために、ノイズ対策等、実装面での配慮が必要になります。

シリアル出力の ADC ですと、ADC 部分とインターフェース部分を比較的少ない本数の制御線でつなぐだけです。また、間にフォト・カプラ等を付けることで、

コンピュータ側と絶縁することも可能ですので、ノイズの影響は少なくてすみます。

測定範囲が 0~12 V の場合、8 ビットでは 47 mV 程度の分解能で多少のノイズも問題ありませんが、12 ビットでは 3 mV の分解能を持ちますので、拡張基板上にアナログ回路を持ち込むのは難しそうです。

◆ パラレル・インターフェースとシリアル・インターフェース

ADC が変換したデータを CPU へ送るためのインターフェースとして、パラレル・インターフェースとシリアル・インターフェースの 2 種類があります。

図 5-1 のパラレル・インターフェースは、CPU へパラレルで変換データを送ります。一般の CPU 等に接続することを前提に作られていますので、通常の CPU 周辺 LSI と同様に、I/O ポートとして動作し、データ・バスを繋ぐだけの簡単な設計で作れます。

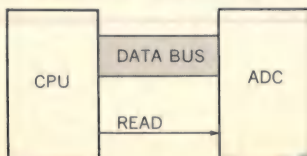
いっぽう図 5-2 に示すシリアル・インターフェースは、DSP やワンチップ CPU 等の、拡張用バスを持たないプロセッサに向いていて、プロセッサ内部にシリアル入力ポートを持っていれば、少ない結線数で接続できます。

◆ AD7890 について

今回使用した AD7890 は、変換時間 5.9 μ s、12 ビットの分解能を持つ ADC で、8 チャンネルのマルチプレクサ、トラック・ホールド・アンプ (サンプル&ホールド)、リファレンス等を内蔵しています。インターフェースは、シリアル形式で最高 10 MHz でデータを転送できます (写真 5-1)。

AD7890 は、「自己クロック (マスタ) モード」と

〈図 5-1〉
パラレル・インターフェースの ADC



〈図 5-2〉
シリアル・インターフェースの ADC

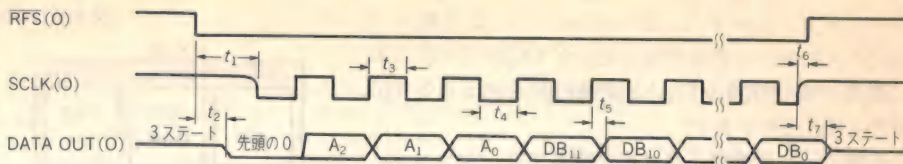


〈写真 5-1〉
AD7890



〈図 5-3〉

自己クロック(マスタ)モードの出力レジスタの読み出しタイミング



「外部クロック(スレーブ)モード」を持ちます。

マスタ・モードは、データ転送のための信号を自分自身が発生するので比較的簡単にインターフェース回路が設計できます。ADC に与える信号は変換開始信号だけになります。クロックには、ADC が変換用クロックに使用する 2.5 MHz を使用するため、転送速度がやや遅くなってしまうのが欠点です。実質的な最高サンプリング周波数は 78 kHz になってしまいます。

スレーブ・モードは、外部からデータ転送に必要な制御信号を ADC へ与えて動作させるモードです。多くの制御信号を与えなくてはならないので、インターフェースをロジック回路で作ると設計が大変です。しかし、変換用クロックとは別に、データ転送用のクロックを与えられ、そのクロック・タイミングを自由に变化させられるので、マイクロプロセッサや DSP からソフトウェア制御をするには向いています。また、データ転送用クロックに最高 10 MHz まで使用できるために、データ転送時間が少なく済みます。実質的なサンプリング周波数は 117 kHz (127 kHz) まで可能です。

AD7890 のタイミング

送出されるデータは図 5-3 のように、変換データ $D_0 \sim D_{11}$ の 12 ビットに加え、チャンネル情報が $A_0 \sim A_2$ が 3 ビットに、先頭のダミー・ビットが加わり全部で 16 ビット構成です。

また、ADC に対して、「シリアル・データ(コマンド)を送ることで、「使用チャンネル指定(3 ビット)、ソフトウェアに変換開始(1 ビット)、スタンバイ・モード(1 ビット)」の指示も可能です(図 5-4)。

今回はソフトウェアからの、変換開始スタンバイ・モード指示は行いませんでした。

マスタ・モードでは、16 ビットのデータを 2.5 MHz のクロックで送りますので、6.4 μs かかります。この

〈図 5-4〉 制御レジスタのビット・アサイン

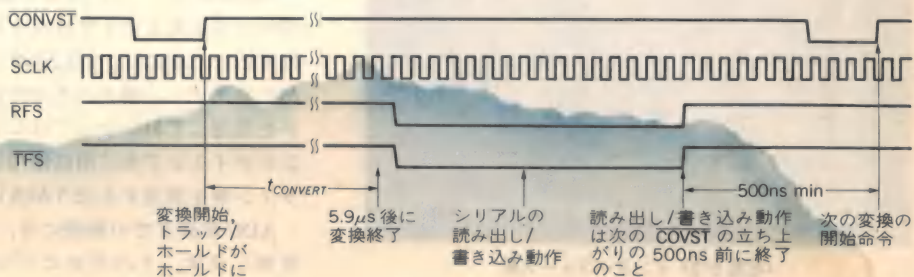
MSB

	A_2	A_1	A_0	CONV	STBY
ビット	説 明				
A_2	アドレス入力。この入力はマルチプレクサのチャンネル選択アドレスの最上位ビット(MSB)				
A_1	アドレス入力。この入力はマルチプレクサのチャンネル選択アドレスの上位 2 番目のビット				
A_0	アドレス入力。この入力はマルチプレクサのチャンネル選択アドレスの最下位ビット(LSB)。アドレスが制御レジスタに書き込みされると内部パルスが発生する。この内部パルス幅は C_{EXT} 端子に接続されたコンデンサの容量値によって決まる。この内部パルスがアクティブ期間中には変換動作は開始されない。このことにより、マルチプレクサのセトリグ時間とトラック・ホールドのアクイジション時間だけトラック・ホールドがホールド・モードになり、変換の開始がウェイトされる。また MUX OUT 端子と SHA IN 端子の間にアンチエイリアシング・フィルタを接続する応用では、SHA IN に加わった信号がサンプリングされるタイミングに、フィルタのセトリグ時間も加えることができる。内部パルスがタイムアウトするとトラック・ホールドがホールド・モードとなり、変換が開始される。				
CONV	変換開始。このビットに 1 を書き込むと、CONV ST 入力と同様に変換が開始される。このビットが 1 の場合でも、連続的な変換動作は行われない。このビットが 1 の場合には書き込み動作の 6 番目のシリアル・クロック・サイクル後に内部パルスと変換が開始する。このビットが 1 の場合にはハードウェアによる変換開始入力(CONVST)は禁止される。このビットに 0 を書き込むと CONVST 入力がいネーブルされる。				
STBY	スタンバイ・モード入力。このビットに 1 を書き込むと、デバイスはスタンバイ(パワーダウン)モードになる。このビットに 0 を書き込むと、デバイスは通常の動作モードとなる。デバイスは SCLK の 7 番目の立ち上がりエッジまでスタンバイ・モードに入らない。このため、デバイスをスタンバイ・モードにする場合にはシリアル書き込み動作で 7 個のシリアル・クロック・パルスが必要				

ため、変換時間と合わせて 12.3 μs 必要になります。さらに、データの読み出し終了直後から、0.5 μs 以内は、次の変換を開始してはいけませんので(これが守られないと、AD7890 の性能を劣化させることがある)、合計 12.8 μs (78 kHz) となります。

〈図 5-5〉

外部クロック(スレーブ)モードで最適性能を得る動作タイミング



スレーブ・モードは、16ビットのデータを最大10 MHzのクロックで送りますので、 $1.6\mu\text{s}$ です。これに変換の待ち時間($0.5\mu\text{s}$)と変換時間を加えると $8\mu\text{s}$ (125 kHz)となります。

実際には、変換終了後次の変換までに、トラック・ホールドのアクイジション時間(トラック・ホールドが確定する時間)の $2\mu\text{s}$ 間おこなうてはなりません、先のデータ転送時間のほうが長くなるので問題ありません。また、図5-5に示すように変換データの転送中に、同時にマルチプレクサのチャンネルを変更する場合は、このチャンネル変更終了後から、アクイジション時間を置かなければなりません。この場合は、チャンネル変更にSCLKが6クロック(10 MHzならば $0.6\mu\text{s}$)かかりますので、 $2\mu\text{s}+0.6\mu\text{s}+$ 変換時間で、 $8.5\mu\text{s}$ (117 kHz)となります。

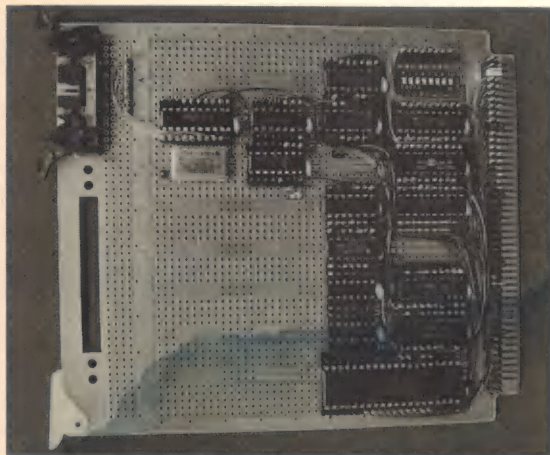
■ 全体の構成

全体の構成は図5-6のようになっています。拡張基板はCPUインターフェース部分のみで、ADC側とはフラット・ケーブル(シールド・ケーブルが良い)で接続します。ADCのインターフェースにはフォト・カプラを入れてノイズ対策すると良いのですが、今回は電源を分離するだけにしました。

図5-7(a)、(b)に全回路を示します。

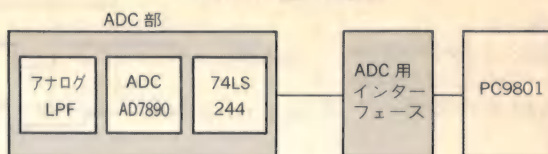
インターフェースはPC98の拡張スロットに入れるために、ロジック(TTL)で組みますので設計が楽な「マスタ・モード」を使用しました。変換速度が落ちますが、目的の10 kHz程度の波形観測はできそうです。

ADCからの変換データは100 kHz近い転送レートで行われます。速度の遅いCPUですと、DMA転送しないと間に合わない速度です。しかし、ADCのデータが12(16)ビットであるために、ノーマル・モードのPC98では、一度にDMA転送できません。また、2回に分けると、タイミング生成が難しくなります。



〈写真5-2〉 インターフェース部

〈図5-6〉 全体の構成



また、DMA転送ですと、一度に64Kバイトまでのデータしか送れないために、長時間のサンプリングも難しくなります。

今回は、ある程度高速なCPUを前提として、タイマ割り込みでサンプリングしています。割り込み処理は、とても時間がかかるのですが、遅いCPUでも、割り込み間隔を延ばすことで、自由にサンプリング・ゲートを遅くして対応できます。

■ インターフェース部分

シリアルとパラレルの変換部分には、シフトレジスタの74LS299(U5、U6)を使用します。これ一つだけでシリアル8ビットの受信と送信ができますので、二つ直列にすることで16ビットのデータを扱います。

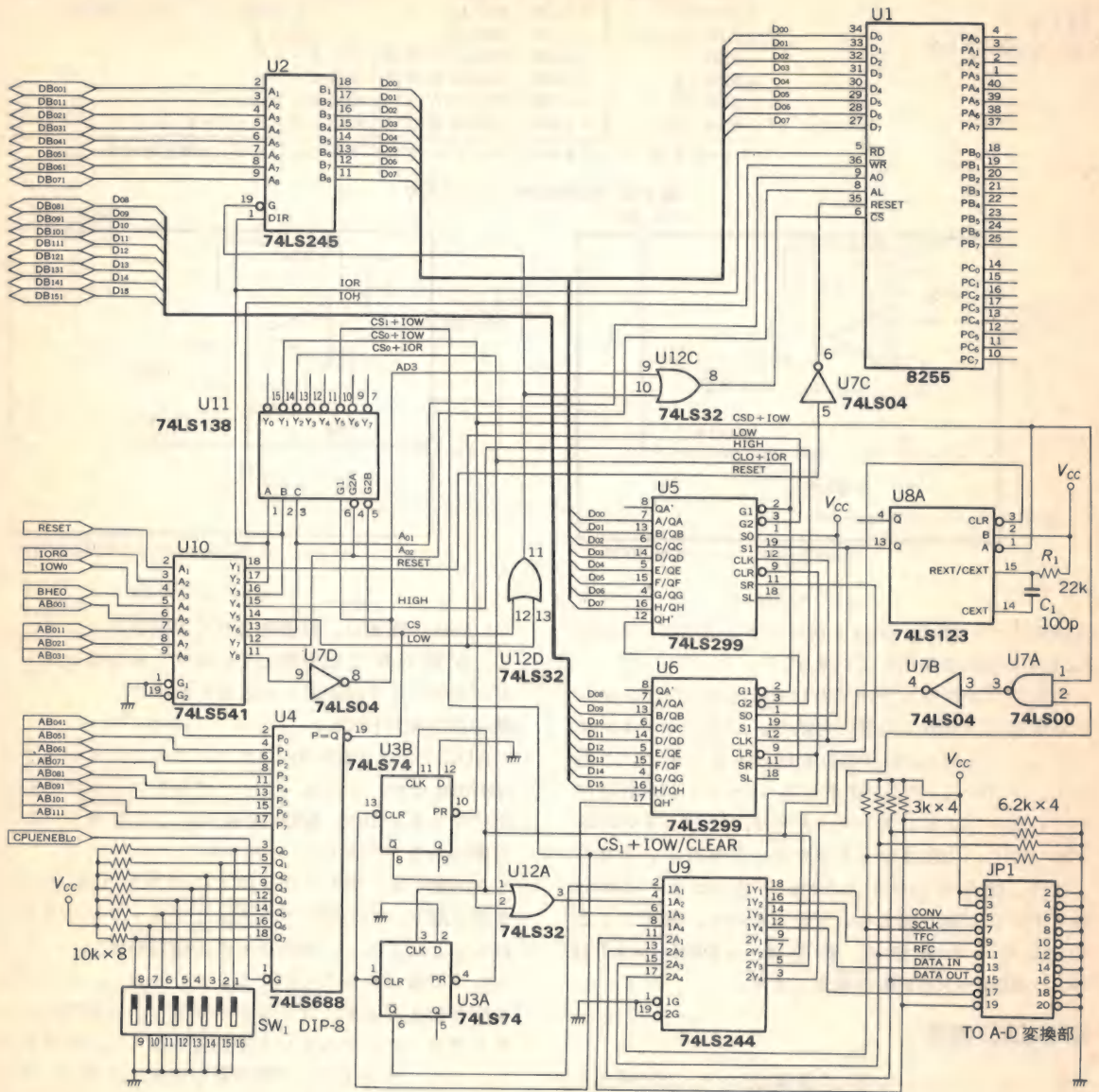
制御信号のタイミング信号は、ADC側から供給されるので、インターフェース部分は比較的簡単にできますが、読み取りのソフトウェア処理の負担を軽くするために、いくつか付加回路を付けました。

- (1) 変換データを読み出すと同時に、次のサンプリング開始信号が自動的に送出されます。74LS74(U3A)を使用し、データ読み出しでCONVSTを“H”(変換開始)に、RFSが“H”(変換データ送出開始)になったら、CONVSTを“L”にします。
- (2) ADCへのコマンド送出も、送りたいデータを書き込むだけで、自動的に送出制御信号を作ります。この場合も、74LS74(U3B)を使用し、データ書き込みでTFS(コマンド送出開始)を“L”にします。RFSが“H”になったら、同時にTFSも“H”にします。
- (3) コマンドの送出は、74LS299に送出コマンドを書き込み、A-D変換データ読み込みと同時に、自動的にコマンド送出をしています。受信のために送られてきたSCLKのクロックで74LS299に書かれたデータがシリアル変換されて出ていきます。もちろん、入れ替わりでA-D変換データが74LS299へ入ってきます。

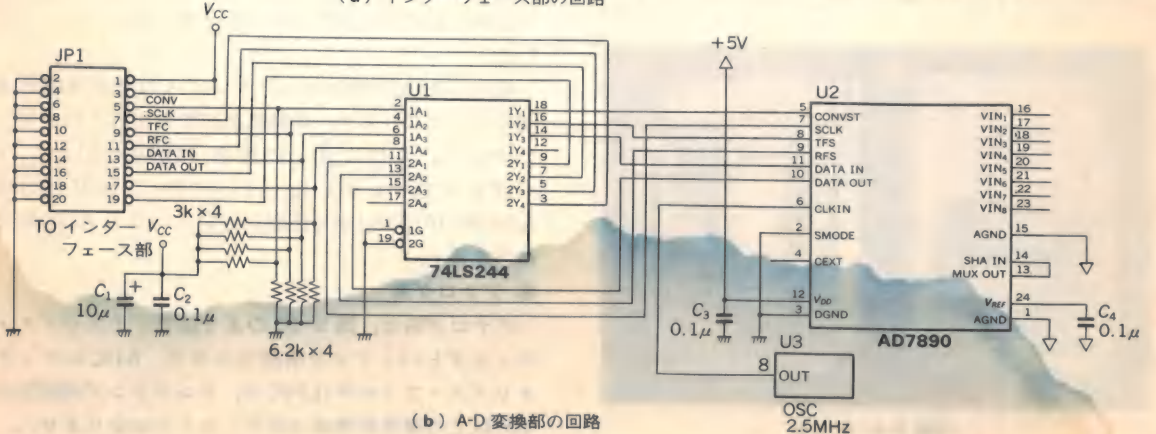
74LS299にデータを書き込むには、74LS299の動作モードを変えなくてはなりません。また、データを書き込むためには、74LS299にクロックを与える必要がありますし、書き込みが完全に終了するまで、モードを固定しておかなくてはなりません。74SL123は、このタイミング生成用に使用しましたが、ディレイ・ライン等を使用するほうが良いかもしれません。

ADC部分までの接続には、フラット・ケーブル等を使用します。そのためにバッファ用として74LS244

〈図 5-7〉 全体の回路



(a) インターフェース部の回路



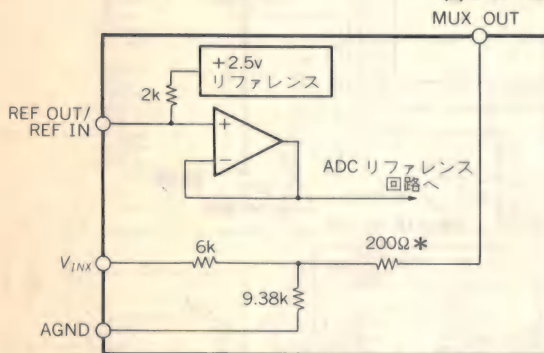
(b) A-D 変換部の回路

〈図 5-8〉
ADC の I/O アドレス

DATA READ	××0H	READ	変換されたデータの読み出し(16 ビット幅)
CH SET	××1H	READ	チャンネルの指定(ビット 5~7 を使用)
ADC RESET	××2H	WRITE	ADC 制御用 IC のリセット
8255 PA	××8H	READ/WRITE	8255 ポート A
8255 PB	××AH	READ/WRITE	8255 ポート B
8255 PC	××CH	READ/WRITE	8255 ポート C
8255 CMD	××EH	READ/WRITE	8255 コマンド・ワード

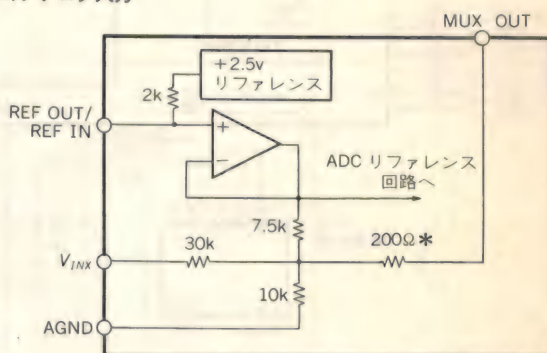
(××は任意に設定できるが、サンプル・プログラムでは「DDH」に固定されている)

〈図 5-9〉 AD7890 のアナログ入力



*マルチプレクサの等価 ON 抵抗

(a) AD7890-4



*マルチプレクサの等価 ON 抵抗

(b) AD7890-10

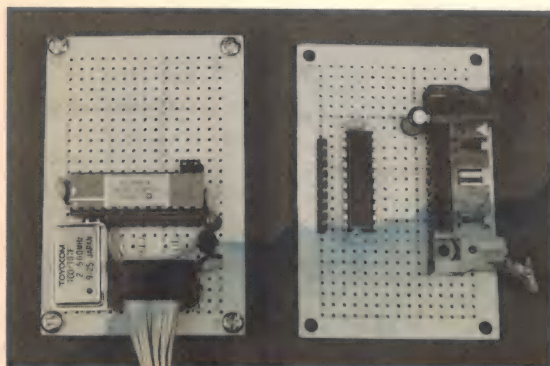
を使用しています。ターミネーションとして、 $3\text{ k}\Omega/6.2\text{ k}\Omega$ の抵抗を使用しています。

今回のプログラムでは使用しませんでした。サンプリングのトリガ入力等の目的で、パラレル・インターフェース LSI ($\mu\text{PD71055}$) を付けてあります。このため、アドレス・デコードやバス・バッファが余計に付いています。アドレス・デコードは、12 ビット分のみデコードし、上位 4 ビットはデコードされていません。

また、図 5-8 に示すように、下位 4 ビットには I/O アドレスが割り当てられていますので、変更できるのは 8 ビット分です。普通は、 $\times\text{DOH} \sim \times\text{DFH}$ か、 $\times\text{EOH} \sim \times\text{EFH}$ を指定します。

ADC 部分

インターフェース部(拡張基板)からの信号は、74LS244 をバッファを通して ADC-LSI へ入ります。



〈写真 5-3〉 ADC 部

74LS244 の電源は、拡張基板から供給されるようにして、ADC の IC とは分離しています。本当はフォト・カプラ等にするのが良いかと思われます。

● ADC (AD7890)

ADC 部分は 2.5 MHz のクロック・ジェネレータと AD7890 です。ここは、ディジタルとアナログの分岐点になりますから、配線(実装)は、よく配慮して行う必要があるでしょう。

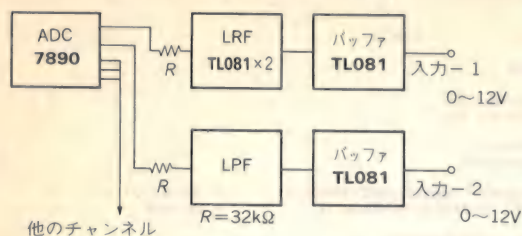
AD7890 は、マルチプレクサの出力と ADC の入力を切り離せ、その間にアンチエリアス・フィルタを入れることによって、複数チャンネル使用時でも一つのフィルタで済ませることができます。しかし、チャンネルを切り替えたとき、フィルタのセトリング時間だけ、サンプリング・タイミングを延ばさなくてはなりません。また、サンプリング周波数を可変した場合、フィルタの定数も変えなくてはならないために、今回はチャンネルごとにフィルタとバッファを入れることにしました。

使用した AD7890-4 のアナログ入力部分は図 5-9 (a) のようになります。単純に抵抗で分圧しているだけです。これにさらに抵抗 ($32\text{ k}\Omega$) を加えて、 $0 \sim 12\text{ V}$ 入力としました。ほかにも、 $\pm 10\text{ V}$ 入力の AD7890-10 [図(b)] や分圧器が付いていない AD7890-2 があります。

● アナログ系

アナログ系は、図 5-10 のようにアンチエリアス・フィルタとバッファで構成されます。ADC のアンチエリアス・フィルタ (LPF) は、サンプリング周波数の半分以上の周波数帯域は阻止しなくてはなりません。

〈図 5-10〉 アナログ系の構成



〈図 5-12〉 ADC サンプリング・プログラム

コマンド名	内 容
SPACE	サンプリング開始
4/6	サンプリングされたデータの左右スクロール表示
2/8	サンプリングされたデータの拡大・縮小(時間軸)
L	保存されたデータ読み出し(FILE)
S	サンプリングされたデータの保存(FILE)
P	自動トリガのためのデータの変動許容率(%)
R	サンプリング時間設定(8253 へのデータ)
J	サンプリングされたデータの指定位置の表示
C	マルチプレクサのチャンネル切り替え
Q	プログラムの終了

分解能(ビット数)が多いほど、その条件は厳しくなってきました。

バッファは、入力インピーダンスを高くするためのもので、OP アンプを非反転増幅器のボルテージ・フォロウ(増幅率=1)で使用します。OP アンプには TL081 を使用しました。

0~12 V 入力の場合は問題ありませんが、マイナスの電圧も必要な場合は、AD7890-10 を使用するか、アナログ回路中にサム・アンプ(加算器)を付けてバイアスをかけます。図 5-11 のようにすれば、いろいろなレンジで使用することもできます。

◆ ADC のソフトウェア

ADC を操作し、データのサンプリング、データの表示のためのプログラムが必要です。

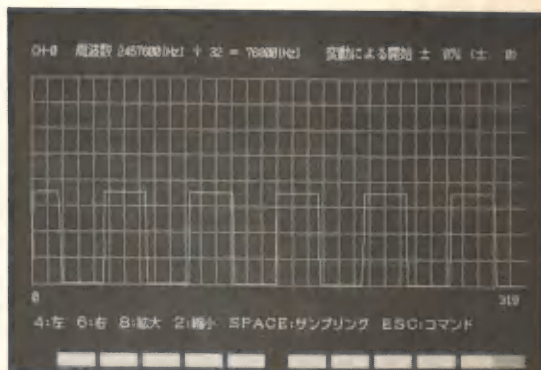
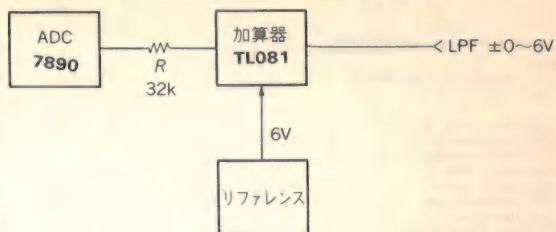
PC9801 のタイマ LSI(8253)のチャンネル#0 を使ったインターバル・タイマで、データのサンプリングを行います。CPU の割り込み処理速度によって調整する必要がありますが、最大 76.8 kHz になります。

8253 のクロックによって、8253 に与える分周比が変わってきます。システム・クロックが 10 MHz 系では、8253 のクロックは 2.4576 MHz となり、76.8 kHz のサンプリング・レートを出すには「32」となります。システム・クロックが 8 MHz 系ならば、19.968 MHz のクロックですから「26」となります。

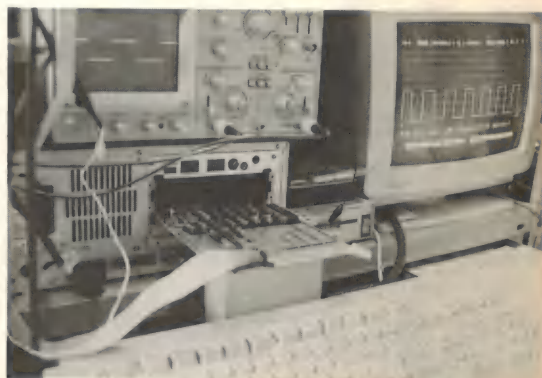
A-D 変換は、DATA READ のリードを行った直後に、自動的に行われます。したがって、ADC RESET のライト後、最低でも 1 回、DATA READ をダミー・リードしなくてはなりません。

また、ADC のマルチプレクサのチャンネルを変える

〈図 5-11〉 マイナス電圧が必要ときの構成



〈写真 5-4〉 ADC サンプリング・プログラムによる表示



〈写真 5-5〉 測定風景

には、CH SET に変更したいデータを書き込みます。チャンネルは 8 個ですから、3 ビット分のデータをビット 5~7 へ入れます。ビット 0~4 までは「0」にしておきます。このチャンネル設定は、DATA READ 後、変換時間以内(5.9 μs)に行わないとなりません。

◆ ADC サンプル・プログラム

今回の実験のために作成したプログラムのリスト ADC.C ADCSUB.ASM を紹介します。ADCSUB.ASM は ADC.C のコンパイル、リンク時に組み込みます。

使用したコンパイラは、Turbo C++ V.1.00、および Turbo ASSEMBLER です(写真 5-4)。


```

/*
**  ADC サンプルングプログラム
**
**  tcc -mc adc.c adcsb.asm
**/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <alloc.h>
#include <dos.h>
#include <conio.h>
#include <pc98.h>
#include <sysstat.h>
#include <io.h>
#include <fontl.h>

#define MAX_DATA 65536
#define MAX_INT 32767

#define TIMER_VEC 0x08
#define TIMER_0 0x71
#define TIMER_MODE 0x77
#define PIC_IMR 0x02
#define PIC_OCW1 0x02
#define PIC_OCW2 0x00
#define EOI 0x20

#define GRPH_X_MAX 639
#define GRPH_Y_MAX 399
#define GRPH_Y_OFF 40

#define ESC 0x1b

#define GDC_STAT 0xa0
#define GDC_CMD 0xa2
#define GDC_PARA 0xa0

#define VECTW 0x4c
#define VECTE 0x8c
#define CSRW 0x49
#define TEXTW 0x78
#define WRITE 0x20
#define START 0x6b

#define BIOS_CRT 0x18

#define MAX_X_BAI 10
#define DATA_RANGE 255

int ssd = 0; ssp = 0;
int sysclock;
int tc;
unsigned int dgd;
int channel = 0;
long freq[2] = {24576001, 19968001};
char help[2][80] =
{
    "4:左 6:右 8:拡大 2:縮小 SPACE:サンプリング ESC:コマンド ",
    "L:読み S:書き P:変動率 R:カウント値 J:ジャンプ C:チャンネル Q:終了"
};
int _fmode = 0_BINARY;

void printstat(void);
void load(char *fname, char huge *buff, long tfrsize);
void save(char *fname, char huge *buff, long tfrsize);
void hst(char *buff);
void command(char *buff);
void oomj(char *buff);
void delchar(char *buff, int point, int *len);

void drawmemo(int xmemo, int ymemo);
void dispdata(char huge *buff, int xscl, long datatop, long datanum);
void drawdata(void);
void drawanadata(int zy, int xscl, int *odat, int ndat, int dnum);

void setlinestyle(unsigned int pat);
void line(int xl, int yl, int x2, int y2, int plane);
void setgdc(int *cmd);
void outgdc(int port, int data);
void dispvram(int on_off);
void crtinit(void);
void setpalet(int *pal);

void clearvram(int sy, int cl, int plane);
void vline(int min, int max, int xx, unsigned int pattern, unsigned int dgd);
void sampstart(char huge *buff, int tcount, int dd, unsigned int dgd, int ch);

int main()
{
    char huge *databuff;
    int key;
    int endflg = 0;
    int disptop, dispmax;
    long cmdbuff[80];
    char tcount[2] = {32, 26};
    int xx = 6; /* 隣りのデータの間の数 */
    int xbai[MAX_X_BAI] = {-50, -20, -10, -5, -2, 1, 2, 5, 10, 20};
    int xmemo = 20; /* メモリ横方向ドット数 */
    int ymemo = 32; /* メモリ縦方向どつと数 */

```

```

    int drawflg;

    long l;

    disptop = 0;
    dispmax = 64;

    crtinit();
    dispvram(1);

    sysclock = (peekb(0, 0x0501) & 0x80) >> 7; /* システムクロック */
    dgd = (peekb(0, 0x054d) & 0x04) << 4; /* GDC clock */

    tc = tcount[sysclock];

    _setcursortype(_NOCURSORS);
    clrscr();
    if ((databuff = faralloc(MAX_DATA)) == NULL) {
        printf("メモリーが確保できません\n");
        exit(1);
    }

    for (l = 0; l < MAX_DATA; l++) *(databuff+l) = 0;

    setlinestyle(0xffff);
    drawmemo(xmemo, ymemo);

    gotoxy(1, 22); cprintf("%s", help[0]);

    while (endflg == 0) {
        drawflg = 0;
        printstat();
        key = getch();
        switch(key) {
            case ESC:
                command(cmdbuff);
                switch(cmdbuff[2]) {
                    case 'S': /* セーブ */
                        save(cmdbuff+3, databuff, MAX_DATA);
                        break;
                    case 'L': /* ロード */
                        load(cmdbuff+3, databuff, MAX_DATA);
                        drawflg = 1;
                        break;
                    case 'P': /* 開始変位の設定 */
                        hst(cmdbuff+3);
                        drawflg = 1;
                        break;
                    case 'R': /* サンプリング周波数の設定 */
                        tc = atoi(cmdbuff+3);
                        break;
                    case 'J': /* 表示位置ジャンプ */
                        disptop = atoi(cmdbuff+3)*2;
                        drawflg = 2;
                        break;
                    case 'C': /* ADC チャンネル変更 */
                        channel = atoi(cmdbuff+3);
                        if ((channel < 0) || (channel > 7)) channel = 0;
                        break;
                    case 'Q': /* 終了 */
                        endflg = 1;
                        break;
                    default: /* 再表示 */
                        drawflg = 2;
                        clrscr();
                        gotoxy(1, 22); cprintf("%s", help[0]);
                        printstat();
                        break;
                }
                break;
            case ' ':
                sampstart(databuff, tc, ssd, dgd << 8, channel);
                drawflg = 1;
                break;
            case '6': /* 右半ページスクロール */
                disptop += dispmax;
                if (disptop >= MAX_DATA) {
                    disptop = MAX_DATA - dispmax*2;
                }
                drawflg = 1;
                break;
            case '4': /* 左半ページスクロール */
                disptop -= dispmax;
                if (disptop < 0) disptop = 0;
                drawflg = 1;
                break;
            case '8': /* 拡大 */
                xx++;
                if (xx >= MAX_X_BAI) xx = MAX_X_BAI-1;
                else drawflg = 2;
                break;
            case '2': /* 縮小 */
                xx--;
                if (xx < 0) xx = 0;
                else drawflg = 2;
                break;
        }
    }

    if (drawflg != 0) {
        if (xbai[xx] < 0) {

```

```

        dispmax = (GRPH_X_MAX*(-xbai[xx]));
        xmemo = 10;
    }else{
        dispmax = (GRPH_X_MAX/xbai[xx]);
        xmemo = xbai[xx]*10;
    }
    disptop &= 0xffff; /* 偶数のみ有効 */
    if (drawflg > 1) drawmemo(xmemo, ymemo);
    dispdata(datbuff, xbai[xx], disptop, dispmax);
}

dispvram(0); /* グラフィック非表示 */
clrscr(); /* テキスト消去 */
_setcursortype(_NORMALCURSOR); /* カーソル点滅 */
return 0;
}

/*
** ステータス表示
*/
void printstat(void)
{
    gotoxy(1, 1);
    printf("CH%d 周波数 %ld[Hz] ÷ %d = %ld[Hz] 変動による開始 ±%3u% (±%4u)", channel, freq[sysclock],
    tc, freq[sysclock]/tc, ssp, ssd);
}

/*
** ロード
*/
void load(char *fname, char huge *buff, long tfrsize)
{
    int fd, errflg, readsize;
    long ptr;

    errflg = 0; ptr = 0;
    if ((fd = open(fname, O_RDONLY|O_BINARY)) != -1) {
        do {
            if (tfrsize < MAX_INT) readsize = (int)tfrsize;
            else readsize = MAX_INT;
            if (read(fd, (char *) (buff+ptr), readsize) == -1) {
                errflg = 1;
                break;
            }
            tfrsize -= readsize;
            ptr += readsize;
        } while (tfrsize > 0);
    }else{
        errflg = 1;
    }

    if (errflg != 0) {
        gotoxy(1, 23); printf("disk read error.\n");
    }
}

/*
** セーブ
*/
void save(char *fname, char huge *buff, long tfrsize)
{
    int fd, errflg, writesize;
    long ptr;

    errflg = 0; ptr = 0;
    if ((fd = creat(fname, S_IWRITE)) != -1) {
        do {
            if (tfrsize < MAX_INT) writesize = (int)tfrsize;
            else writesize = MAX_INT;
            if (write(fd, (char *) (buff+ptr), writesize) != writesize) {
                errflg = 1;
                break;
            }
            tfrsize -= writesize;
            ptr += writesize;
        } while (tfrsize > 0);
    }else{
        errflg = 1;
    }

    if (errflg != 0) {
        gotoxy(1, 23); printf("disk write error.\n");
    }
}

/*
** サンプリング開始待ち変動量を%単位で指定する
*/
void hst(char *buff)
{
    int per, max;

    per = atoi(buff);
    if ((per > 0) && (per < 100)) {
        max = 0xffff; /* 12ビット最大値 */
        max = (max/100)*per;
    }
}

```

```

        ssp = per;
        ssd = max;
    }else{
        ssp = 0;
        ssd = 0;
    }
}

/*
** コマンド入力
*/
void command(char *buff)
{
    gotoxy(1, 22); printf("%s", help[1]);
    gotoxy(1, 23); clrscr();
    _setcursortype(_NORMALCURSOR);
    printf(">>>");
    buff[0] = 70;
    cgets(buff);
    _setcursortype(_NOCURSOR);
    oomoji(buff+2);
    gotoxy(1, 23); clrscr();
    gotoxy(1, 22); printf("%s", help[0]);
}

/*
** 小文字大文字変換
*/
void oomoji(char *buff)
{
    int i, len;

    len = strlen(buff);
    for (i = 0; i < len; i++) {
        if (buff[i] == ' ') delchar(buff, i, &len); /* スペースは削除 */
        if (((buff[i]>0x80) && (buff[i]<0xA0)) || ((buff[i]>0xdf) && (buff[i]<0xfd))) {
            i++;
        }else{
            if (buff[i] < 'a') buff[i] = (char)0;
            if ((buff[i] >= 'a') && (buff[i] <= 'z')) {
                buff[i] = buff[i] - ('a'-'A');
            }
        }
    }
}

/*
** 文字列から1文字削除
*/
void delchar(char *buff, int point, int *len)
{
    int i;

    for (i = point; i < (*len); i++) {
        buff[i] = buff[i+1];
    }
    (*len)--;
}

/*
** データを画面に表示する。
*/
void dispdata(char huge *buff, int xscl, long datatop, long datanum)
{
    long i;
    int j, di;
    int odat, dat;
    int min, max;
}

```


<リスト 5-1> ADC サンプルング・プログラム (つづき)

```

clearvram(GRPH_Y_OFF, DATA_RANGE+1, 1);

if (xscl < 0)      di = (-xscl);
else              di = 1;

for (i = 0; i <= datanum; i += di) {
    if (xscl > 0) { /* 通常折れ線グラフモード */
        dat = ((buff[datatop+i*2+1]<<4) | (buff[datatop+i*2]>>4))&0xff;
        drawanadata(GRPH_Y_OFF, xscl, &dat, (int)i);
    } else { /* 最大値最小値モード */
        min = DATA_RANGE; max = 0;
        for (j = 0; j < di; j++) {
            dat = ((buff[datatop+(i+j)*2+1]<<4) | (buff[datatop+(i+j)*2]>>4))&0xff;
            if (dat < min) min = dat;
            if (dat > max) max = dat;
        }
        vline(min, max, (int)(i/di), 0xffff, dgd<<8);
    }
}
gotoxy(1, 20);  cprintf("%-7lu", datatop/2);
gotoxy(73, 20); cprintf("%7lu", datatop/2+datanum);
}

/* メモリを描く
**
*/
void drawmemo(int xmemo, int ymemo)
{
    int x, y, dx, m10, tenflg, count;
    unsigned lpat[2] = {0xffff, 0xeccc};

    clearvram(GRPH_Y_OFF, DATA_RANGE+5, 0);

    if (xmemo > 20) {
        dx = xmemo/2;
        tenflg = 2;
    } else {
        dx = xmemo;
        tenflg = 0;
    }

    /* X方向にメモリを描く */
    count = 0;
    for (x = 0; x <= GRPH_X_MAX; x+=dx) {
        if (tenflg != 0) { /* 間隔が広い場合点線を入れる */
            setlinestyle(lpat[count%2]);
        }
        if ((count%10) == 0) m10 = 4; /* 10本おきに長く */
        else m10 = 0;
        line(x, GRPH_Y_OFF, x, GRPH_Y_OFF+DATA_RANGE+m10, 1);
        count++;
    }

    /* Y方向にメモリを描く */
    setlinestyle(lpat[0]);
    for (y = GRPH_Y_OFF; y <= GRPH_Y_OFF+DATA_RANGE+1; y+=ymemo) {
        line(0, y, GRPH_X_MAX, y, 1);
    }

    /* データを描く (アナログ用)
    **
    ** IN
    ** zx:   O点X座標 (ドット)
    ** zy:   O点Y座標 (ドット)
    ** xscl: X方向隣りの点とのドット数
    ** yscl: Y方向1あたりのドット数x10
    **
    */
    void drawanadata(int zy, int xscl, int *odat, int ndat, int dnum)
    {
        int xx, yy;

        ndat = DATA_RANGE-(ndat); /* データを画面表示用に逆転 */
        if (dnum > 0) {
            xx = dnum*xscl; /* X方向1つ前のデータのドット計算 */
            yy = zy; /* Y方向基準ドットの計算 */
            line(xx-xscl, yy+(*odat), xx, yy+ndat, 2);
        }
        *odat = ndat;
    }

    /*
    ** GDCのラインスタイルの設定
    **
    */
    void setlinestyle(unsigned int pat)
    {
        int cmd[16];

        cmd[0] = WRITE;
        cmd[1] = -1;
        setgdc(cmd);

        cmd[0] = TEXTW;
        cmd[1] = pat&0xff;
        cmd[2] = pat>>8;
        cmd[3] = -1;
        setgdc(cmd);
    }

    /*
    ** GDCで直線を引く
    **
    */
    void line(int x1, int y1, int x2, int y2, int plane)
    {
        int dx, dy, dmy, cmd[16];
        unsigned int dir, dc, d, d2, d1, ead;

        if ((x1 > x2) || ((x1 == x2) && (y1 > y2))) {
            dmy = x1; x1 = x2; x2 = dmy;
            dmy = y1; y1 = y2; y2 = dmy;
        }
        dx = x2-x1; dy = y2-y1;

        if (dy > 0) {
            if (dx < dy) {
                dir = 0;
                dmy = dx; dx = dy; dy = dmy;
            } else {
                dir = 1;
            }
        } else {
            dy = -dy;
            if (dx > dy) {
                dir = 2;
            } else {
                dir = 3;
                dmy = dx; dx = dy; dy = dmy;
            }
        }
        dc = dx; d1 = 2*dy; d = d1-dx; d2 = d-dx;

        cmd[0] = VECTW;
        cmd[1] = 0x08|dir;
        cmd[2] = dc&0xff;
        cmd[3] = (dc>>8)|dgd;
        cmd[4] = d&0xff;
        cmd[5] = d>>8;
        cmd[6] = d2&0xff;
        cmd[7] = d2>>8;
        cmd[8] = d1&0xff;
        cmd[9] = d1>>8;
        cmd[10] = 0;
        cmd[11] = 0;
        cmd[12] = -1;

        while ((inportb(GDC_STAT)&0x08) != 0) { /* 描画が終了するまで待つ */
            outportb(0x5f, 0);
        }

        setgdc(cmd);

        ead = plane*0x4000+y1*40+x1/16;
        cmd[0] = CSRW;
        cmd[1] = ead&0xff;
        cmd[2] = ead>>8;
        cmd[3] = (x1%16)<<4;
        cmd[4] = -1;
        setgdc(cmd);

        cmd[0] = VECTE;
        cmd[1] = -1;
        setgdc(cmd);
    }

    /*
    ** G-GDCに対してコマンド+パラメータ列を送る.
    **
    */
    void setgdc(int *cmd)
    {
        int i = 1;

        outgdc(GDC_CMD, cmd[0]);
        while (cmd[i] != -1) {
            outgdc(GDC_PARA, cmd[i]);
            i++;
        }
    }

    /*
    ** G-GDCに対して1バイトのデータを送る
    **
    */
    void outgdc(int portadr, int dat)
    {
        while ((inportb(GDC_STAT)&0x02) != 0) { /* FIFOが空くまで待つ */
            outportb(0x5f, 0);
            outportb(0x5f, 0);
        }
        outportb(portadr, dat);
        outportb(0x5f, 0);
    }

```

```

/*
** パレットを設定する。
*/
void setpalet(int *pal)
{
    int i;
    int padr[4] = {0xaa, 0xaa, 0xac, 0xa8};

    for (i = 0; i < 4; i++) {
        outportb(padr[i*4], (pal[i]<<4)|pal[i*4]);
    }
}

/*
** VRAM表示ON/OFF
*/
void dispvram(int on_off)
{
    union REGS regs;

    regs.h.ah = 0x40 + (on_off);

    int86(BIOS_CRT, &regs, &regs);
}
/*

```

```

** 8色 400ライン
** VRAM消去 パレット設定
** 描画VRAM表 表示VRAM表
*/
void crtinit(void)
{
    int i;
    union REGS regs;
    int pal[] = {0, 1, 6, 6, 4, 5, 6, 7};

    regs.h.ah = 0x42;
    regs.h.ch = 0xc0;

    int86(BIOS_CRT, &regs, &regs); /* 400ライン, カラー, */
    int86(BIOS_CRT, &regs, &regs);

    outportb(0x6a, 0); /* 8色モード */
    outportb(0x5f, 0);
    outportb(0xa4, 0); /* 表示画面 VRAM表 */
    outportb(0x5f, 0);
    outportb(0xa6, 0); /* 描画画面 VRAM表 */

    for (i = 0; i < 3; i++) clearvram(0, 400, i);

    setpalet(pal);
}

```

リスト 5-2> ADC サンプルング・プログラム (アセンブラ部)

```

;
; ADCサンプルングプログラムアセンブラ部
;
; 186命令使用為 V30以降有効
;
PUBLIC _clearvram ; VRAM消去
PUBLIC _vline ; 最大値-最小値 直線描画
PUBLIC _sampstart ; サンプルング開始

ADC_DATA equ 0dd0h
ADC_CH equ 0dd1h
ADC_RESET equ 0dd2h

GRPH_Y_OFF equ 40
GDC_RED equ 8000h

GDC_STAT equ 0a0h
GDC_CMD equ 0a2h
GDC_PARA equ 0a0h

GDC_VECTW equ 4ch
GDC_VECTE equ 6ch
GDC_CSRW equ 49h
GDC_TEXTW equ 78h

LOCALS
.186
.MODEL COMPACT
.DATA
.DATA?

maskreg db ? ; 8259マスタ IMR
timerseg dw ? ; インターバルタイマ旧ベクタ
timeroff dw ?
min db ? ; 最小値
max db ? ; 最大値
xx dw ? ; X座標 (0~639)
mul40 db ?
div16 db ?

.CODE

;
; VRAM消去
;
void clearvram(int sy, int cl, int plane)
; sy : 消去開始Y座標
; cl : 消去行数
; plane : VRAMアレーン

_clearvram PROC
    push bp
    mov bp, sp
    push es
    push di

    call checkdraw ; 描画終了まで待つ

    mov bx, [bp+4] ; 消去開始行
    mov ax, 80
    mul bx
    mov di, ax

    mov bx, [bp+6] ; 消去行数
    mov ax, 40
    mul bx
    mov cx, ax

```

```

    mov bx, [bp+8] ; 消去VRAMアレーン
    mov ax, 800h
    mul bx
    add ax, 0a800h
    mov es, ax

    cld
    mov ax, 0
    rep stosw

    pop di
    pop es
    pop bp
    ret

_clearvram ENDP

;
; 最大値-最小値に垂直方向に線を引く
;
void vline(int min, int max, int xx, unsigned linestyle, unsigned int dgd)
; BX CX DX DI ES はサンプルング部が使用している為変更してはいけない。
;
_vline PROC
    push bp
    mov bp, sp
    push si

    call checkdraw ; 描画終了まで待つ

    mov al, GDC_TEXTW ; ラインスタイルの設定
    call outcldb
    mov ax, [bp+10]
    call outparaw

    mov al, GDC_VECTW
    call outcldb
    mov al, 08h ; 直線, 描画方向O
    call outparab ; set SL, R, C, T, L, DIR
    mov ax, [bp+6]
    sub ax, [bp+4] ; ax = max-min
    or ax, [bp+12] ; ax = ax | dgd
    call outparaw ; set DC
    neg ax ; ax = -(max-min)
    call outparaw ; set D
    rol ax, 1 ; ax = ax*2
    call outparaw ; set D2
    mov ax, 0
    call outparaw ; set D1

    mov al, GDC_CSRW ; 描画アドレスの設定
    call outcldb
    mov ax, 255
    sub ax, [bp+6] ; ax = 255-max
    mov [mul40], 40
    mul [mul40] ; ax = max * 40
    mov si, ax ; ax を一時退避
    mov ax, [bp+8] ; ax = xx
    mov [div16], 16
    div [div16] ; ax = xx/40
    push ax
    mov ah, 0
    add ax, si
    add ax, GDC_RED+GRPH_Y_OFF*40
    call outparaw ; set EAD
    pop ax
    mov al, ah ; xx/40 の余り
    shl al, 4
    call outparab ; set dAD

```



```

mov     al,GDC_VECTE    ; 描画開始
call    outcldb
pop     si
pop     bp
ret
_vline ENDP
;
; G-GDC 描画終了まで待つ
;
checkdraw:
push    ax
@@notready:
out     5fh,al
out     5fh,al
in      al,GDC_STAT
test    al,08h
jnz     @@notready
pop     ax
ret

```

; G-GDC の FIFO が空くまで待つ

```

;
;
checkfifo:
push    ax
@@notready:
out     5fh,al

out     5fh,al
in      al,GDC_STAT
test    al,02h
jnz     @@notready
pop     ax
ret

```

; G-GDC にコマンドを送る

```

IN      al      : GDC に送るコマンド
;
outcldb:
call    checkfifo
out     5fh,al
out     5fh,al
out     GDC_CMD,al
ret

```

; G-GDC に 1 バイトパラメータを送る

```

IN      al      : GDC に送るパラメータ
;
outparab:
call    checkfifo
out     5fh,al
out     5fh,al
out     GDC_PARA,al
ret

```

; G-GDC に 2 バイトのパラメータを送る

```

IN      ax      : GDC に送るパラメータ
;
outparaw:
push    ax
call    outparab
mov     al,ah
call    outparab
pop     ax
ret

```

; サンプリング開始

; void sampstart(char huge *buff,int tc,int dd,unsigned int dgd,int ch)

```

; buff : サンプリングデータ書き込みアドレス
; tc   : サンプリング間隔 (インターバルタイマカウンタ値)
; dd   : = 0 即サンプリング開始
;       : != 0 初期値から ±dd 変動したらサンプリング開始
; dgd  : GDC clock 5MHz -> 01000000b , 2.5MHz -> 0
; ch   : ADC のチャンネル (0-7)
;
_sampstart PROC

```

```

push    bp
mov     bp,sp
push    ds
push    es
push    di
push    si

```

```

call    adreset          : ADC 初期設定
les     di,dword ptr [bp+4] : buff
mov     cx,[bp+8]         : tc
call    settimer         : タイマー登録
mov     ax,[bp+10]        : dd
call    setstart
cld
mov     dx,ADC_DATA       : adc 1/0 アドレス設定
mov     al,0feh           : タイマー割り込み許可
out     02h,al

call    dispdata         : インジケータ表示
call    endtimer         : ベクターを戻す
pop     si
pop     di
pop     es
pop     ds
pop     bp
ret
_sampstart ENDP

```

; ADC リセット、チャンネル設定

```

;
; IN      [bp+14] : ADC channel
; ax, dx 破壊
;
adreset:
mov     dx,ADC_RESET      : ADC reset
out     dx,al
mov     dx,ADC_CH        : ADC channel set
mov     ax,[bp+14]
shl     al,5
out     dx,al
mov     dx,ADC_DATA       : ADC dummy read
in      ax,dx
call    wait13
ret

```

; 13μ秒ウエイト

```

; 破壊 ax
wait13:
mov     ah,26
@@loop:
out     5fh,al           : 0.5μ秒
dec     ah
jnz     @@loop
ret

```

; サンプリング開始待ちの変動範囲の指定

```

;
; IN      : 変動値
; ax      : 変動値
; es:di   : 書き込みバッファ
; OUT     :
; bx      : 最大値
; cx      : 最小値
; bx = cx の場合すぐにサンプリング開始
;
; 破壊 ax

```

```

setstart:
mov     es:[di],0ffffh ; 未読み込みマーク

mov     bx,ax
mov     cx,ax
cmp     ax,0
jz      @@exit

mov     dx,ADC_DATA     : ADC 初期値
in      ax,dx
and     ax,0ffffh
add     bx,ax
sub     ax,cx
jns     @@puls
mov     ax,0
@@puls:
mov     cx,ax
@@exit:
ret

```

; インジケータ表示

```

;
; IN      :
; es:di   : 書き込みバッファ
;

```

```

; 破壊 ax
dispdata:
cmp     es:[di],0ffffh      ; サンプリングが開始
jz      dispdata            ; するまで待つ

mov     [xx],0               ; x方向座標クリアー
@@minmaxstart:
mov     [min],0fffh          ; 最小値データクリア
mov     [max],0              ; 最大値データクリア
call    clearline            ; ライン消去
call    setline              ; ライン描画

mov     ax,[xx]              ; xx をインクリメント
inc     ax
cmp     ax,640               ; 639 越えたら 0 に
jb      @@setcx
mov     ax,0
@@setcx:
mov     [xx],ax

or      di,di                ; サンプリングが終了するまで待つ
jnz     @@minmaxstart
ret

```

描画中に最大値と最小値をチェックする

```

setminmax:
out     5fh,al
out     5fh,al
in      al,GDC_STAT          ; GDCステータス

push    ax
cmp     bx,cx                ; 開始待ち終了?
jnz     @@notstart
mov     ax,es:[di-2]         ; データを読み出す。(サプリング中)
jmp     @@conv256
@@notstart:
mov     ax,es:[di]           ; データを読み出す。(開始待ち)
@@conv256:
shr     al,4                 ; データを12ビットから
shl     ah,4                 ; 8ビットに変換する。
or      al,ah
cmp     al,[min]             ; 最小値かチェック
jae     @@notin
mov     [min],al
@@notin:
cmp     al,[max]             ; 最大値かチェック
jbe     @@notamax
mov     [max],al
@@notamax:
pop     ax

test    al,02h               ; 描画中か?
jnz     setminmax
ret

```

インジケータ用1ライン消去

```

clearline:
call    setminmax            ; dgd
push    [bp+12]              ; ライン消去
push    0                    ; xx
push    [xx]                 ; max
push    0fffh                ; min
push    0

call    _vline               ; スタックを戻す
add     sp,10
ret

```

インジケータ用最小値-最大値描画

```

setline:
call    setminmax            ; dgd
push    [bp+12]              ; ライン描画
push    0fffh                ; xx
push    [xx]                 ; max
mov     al,[max]
mov     ah,0
push    ax
mov     al,[min]             ; min
push    ax
call    _vline               ; スタックを戻す
add     sp,10
ret

```

インターバルタイマーをセットする

```

IN      cx :timer count
破壊   ax,bx,dx

```

```

settimer:
push    ds
push    es
in      al,02h               ; 割り込みマスキレジスタ保存
mov     [maskreg],al
mov     al,0fffh
out     02h,al               ; 全割り込み禁止

mov     ax,3508h             ; インターバルタイマの
int     21h                  ; 旧ベクタを保存
mov     ax,es
mov     [timerseg],ax
mov     [timeroff],bx

mov     ax,cs                ; インターバルタイマに
mov     ds,ax                ; 新ベクタを登録
mov     dx,offset sampling
mov     ax,2508h
int     21h

mov     al,36h               ; タイマー#0モード3
out     77h,al
mov     al,cl                ; カウンター下位設定
out     71h,al
mov     al,ch                ; カウンター上位設定
out     71h,al

pop     es
pop     ds
ret

```

インターバルタイマーを終了する。

```

; ax,dx 破壊
endtimer:
push    ds
mov     al,[maskreg]         ; 割り込みマスクを戻す
or      al,00000001b
out     02h,al
mov     dx,[timeroff]
mov     ax,[timerseg]        ; ベクタを戻す
mov     ds,ax
mov     ax,2508h
int     21h
pop     ds
ret

```

サンプリング部

```

bx      : サンプリング待ち最大値
cx      : サンプリング待ち最小値
dx      : ADCデータ1/Oアドレス
es:di   : 書き込みアドレス
; 高速化の為 上記レジスタは保存していない。

```

```

sampling:
push    ax
in      ax,dx                ; サンプリングデータを読む
cmp     bx,cx                ; サンプリング開始待ち?
jnz     @@check
stosw

or      di,di                 ; di = 0 でサンプリング終了
jz      @@finish

```

```

@@eoi:
mov     al,20h               ; 割り込み終了
out     00h,al
pop     ax
iret

```

```

@@finish:
mov     al,01111001b         ; タイマー割り込み禁止
out     02h,al
@@eoi:

```

```

@@check:
and     ax,0fffh             ; 12ビットデータ
cmp     ax,bx                ; 最大値より大きい?
jg      @@start
cmp     ax,cx                ; 最小値より小さい?
jl      @@start
mov     es:[di],ax
@@eoi:

```

```

@@start:
add     di,2
stosw
mov     cx,bx
jmp     @@eoi
end

```


編集雑記

編集部から

● 休みの日に久しぶりにパソコンのファイルの整理をすることにした。日頃アプリケーションしか立ち上げない家庭用のマシンでは、ハード・ディスクがいっぱいになり、妻や娘がエラー・メッセージに驚かない限り、めったにファイルを覗くことはない。新しいアプリケーションを適当に動くように、いいかげんにインスツールしていると、ハード・ディスクの中身は、ゴミファイルだらけになってしまう。おまけに CON FIG を何度も書き直しているの、なぜこんな記述をしていたのかわからなくなってしまう。

● 思い切って最初からやり直せばよいのだが、100MB 以上もあるハード・ディスクの中身をバックアップする気にもならない。

● 商売がら本や資料も部屋の中に溢れがちで、ほっておくと古紙の山と化し、ある日突然必要なものも不必要なものもすべて資源ゴミに出してしまう。

● ハード・ディスクの中身も全くこれと同じである。最低限必要な辞書ファイルなどをバックアップして、フォーマットしてしまい、ついでに DOS のバージョンアップを図るこ

とにした。

案の定、娘や家内からは文句がでる。せっかく花子で書いた絵が消えてしまったとか、ゲームの得点が出てこない等々である。

● 我が家では、家計簿や住所録などという常にデータのメンテが必要なのは、コンピュータなるものには依存していなかったもので、白い目でみられる程度で、実害はなく(と思っているのは私だけか?)、不法コピーのゲームがなくなってしまった程度で済んだが、これが業務用パソコンともなるとそう簡単には事が運ばない。やはりコンピュータは一人1台か、せめて一人1ハード・ディスクにでもしないと、と思いつつ休日日を1日過ごしてしまった。

● 夕食時に妻曰く「こんなにお天気が良い日に1日パソコンに向き合っているなんて、結構オタクね!」

● 桜の花も満開である!!

● パソコンの話題の中心はすっかり AT に移ってしまっているかに見える。しかし、現在稼働中のマシンの大部分は 98 系といえるだろう。この号では、普及率 No.1 の 98 シリーズをとりあげた。本誌 No.3 で同じテーマを特集してから 7 年が経っており、98 シリーズも進化している。(哲)

● トランジスタ技術 SPECIAL の既刊号で紹介しました基板等の頒布サービスを、申し込み締め切り日を過ぎて受け付けているものがありますのでお知らせします。それらは、No.20 の MICRO-CAP III, No.23 の PAL ライタ基板, PALASM ソフト, No.29 の Z80 マイコン・キット, No.38 の拡張 I/O モジュール・キットです。申し込み方法は各雑誌掲載のとおりです。

● 本誌掲載記事の利用についてのご注意——本誌掲載記事には著作権があり、また工業所有権が確立されている場合があります。したがって、個人で利用される場合以外は所有者の承諾が必要です。

また、掲載された回路、技術、プログラムを利用して生じたトラブルなどについては、小社ならび著作権者は責任を負いかねますのでご了承ください。

● ご質問はお手紙で——本誌掲載記事に関する技術的なご質問は、往復はがきか、返信用封筒を同封した書簡を編集部あてお寄せください。執筆者に回送し、直接回答していただきます。質問の内容は当該記事を逸脱しない範囲で、できる限り具体的に明記してください。また、お電話によるご質問にはお応えできませんので、ご了承ください。

次号のお知らせ(6月29日発売)

特集 高周波回路設計のすべて

このところ、高周波分野の応用はめざましく、携帯電話や通信機だけでなく構内データ伝送やリモート・コントロールの分野での応用が広がっています。高周波の応用はワイヤレスという、他にない特徴があり、今後もその要求が広がっていくと思われます。

次号は、最近技術者が減ったと言われるこの分野を詳解します。

トランジスタ技術 SPECIAL

No. 45

発行所 CQ出版株式会社 (無断転載を禁じます)

〒170 東京都豊島区集鴨 1-14-2

電話 編集部: 03(5395)2125, 広告部: 03(5395)2132

営業部: 03(5395)2141

振替 東京 0-10665 (5月1日から 00100-7-10665 に変わります)

発行人 神戸一夫

編集人 増田久喜

Printed in Japan

© CQ出版株式会社 1994 (定価は表四に表示してあります)

1994年5月1日発行

印刷・製本 三晃印刷株式会社

MS-DOS 基本プログラミング第5集

PC98ユーティリティの作り方



トラ技コンピュータ編集部編 B5変形判 224頁

定価2,300円 送料380円

最新では、コンパイラなども安く入手できますから、プログラムを自分で作ることも容易になってきました。しかし、アルゴリズムを学習して、ある目的のプログラムを作ることはできるようになっても、それを実用的なユーティリティにする方法はいかなかに学習できません。

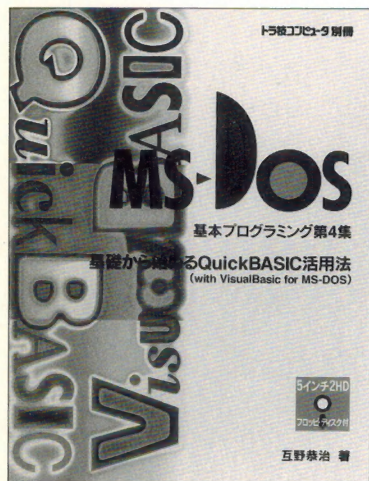
本書はそういったことで悩んでいる読者のために、PC98シリーズで動作するユーティリティの作り方を解説しました。第1部ではMS-DOSの基本的なプログラムの作り方を示し、第2部では常駐型ソフトウェアの作り方について詳しく解説してあります。

MS-DOS 基本プログラミング第4集

基礎から始めるQuickBASIC活用法

(with VisualBasic for MS-DOS)

互野恭治 著 B5変形判 252頁 定価2,400円 送料380円



本書は、トラ技コンピュータの1992年1月号から1992年12月号まで連載された「QuickBASICでエンジョイ・プログラミング」をまとめ、さらに内容を充実させたものです。

加筆した内容は、▶QuickBASICの基本操作について、▶構造化プログラミングについて、▶QuickBASICの文法について、▶ライブラリの作成法について、▶VisualBasicのプログラミングについてなどです。さらに、応用プログラム例を増やし、本書だけでアマチュア・プログラマーが必要とするプログラミング例は、ほとんど網羅してあります。

また、MS-DOS版のVisualBasicで、ほとんどのプログラムが動作することを確認しています。

世の中では、C言語でなければプログラミング言語ではないというような風潮がありますが、市販ソフトウェアを作るプログラマーでなければ、プログラミングの容易さや、わかりやすさなどQuickBASICのほうが優れているといっても過言ではありません。特に、MS-Windows 3.1が主流になると、VisualBasicが一般プログラマーのプログラミング言語の主流になります。

MS-DOS基本プログラミング 第1集

PC9801グラフィックス・プログラミング

トラ技コンピュータ編集部編 B5変形判 238頁 定価2,300円 送料380円

MS-DOS基本プログラミング 第2集

PC9801の割り込みとBIOS活用法

トラ技コンピュータ編集部編 B5変形判 252頁 定価2,300円 送料380円

MS-DOS基本プログラミング 第3集

PC98アセンブラ・プログラミング入門

相沢一石 著 B5変形判 248頁 定価2,400円 送料380円

すべて
5インチ2HD FD付



